НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Кафедра автоматизованих систем обробки інформації і управління

«На правах рукопису»

УДК __004.912_____

«До захисту допущено»
В.о. завідувача кафедри
_____ *О.А.Павлов*
(підпис) (ініціали, прізвище)

"___"_____2020 р.

Магістерська дисертація

121 «Інженерія програмного забезпечення»
зі спеціальності
на тему: « Програмна бібліотека обробки текстової інформації для ApacheSpark»

Виконав: студент VI курсу, групи *ІП-92мп*

Якимчук Олександр Анатолійович _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник *ст. вик. Олійник Юрій Олександрович* _____
(посада, науковий ступінь, , вчене звання, прізвище та ініціали) (підпис)

Консультант *доц., к.т.н., Ліщук К.І.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент *доц., к.т.н., доц. Ткач М.М.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Київ – 2020 року

РЕФЕРАТ

Магістерська дисертація: 76 с., 14 рис, 25 таб., 17 джерел.

Актуальність теми: відсутність інструментів для обробки великих об'ємів текстових даних, що підтримують українську та російську мови.

Мета дослідження:синтез програмного рішення для обробки великих об'ємів текстових даних та підтримкою обробки українськомовних та російськомовних текстів.

Об’єкт дослідження: текстові дані.

Предмет дослідження: обробка текстових даних для української та російської мов з підтримкою розподілених обчислень.

Методи дослідження: у даній дисертаційній роботі застосовувалися методи обробки природної мови, засновані на правилах, словниках та існуючих лінгвістичних ресурсах.

Наукова новизна: створено програмну бібліотеку для технології ApacheSpark на мові Python, яка на відміну від існуючих містить функції реферування та підтримку обробки українськомовних текстів.

Практичне значення отриманих результатів визначається тим, що розроблена програмна бібліотека дозволить автоматизувати обробку текстових даних.

Зв’язок роботи з науковими програмами, планами, темами: робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Програмна бібліотека обробки текстової інформації для ApacheSpark». Державний реєстраційний номер0117U000924.

Апробація: Основні положення роботи доповідались і обговорювались на III всеукраїнській науково-практичній конференції молодих вчених та студентів «Інформаційні системи та технології управління» (ІСТУ-2020)

Публікації: Наукові положення дисертації опубліковані в тезах конференції «ІНФОРМАТИКА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА – ІОТ-2020»

Ключові слова: ОБРОБКА ПРИРОДНЬОЇ МОВИ, ЛЕМАТИЗАЦІЯ, ВЕКТОРИЗАЦІЯ, РЕФЕРУВАННЯ, ТЕКСТОВІ ПОТОКИ ДАНИХ.

ABSTRACT

Master's dissertation consists 76 pages, 14 images, 25 tables, 17 referring sources.

Topicality: lack of tools for processing large amounts of text data that support Ukrainian and Russian.

The purpose of the dissertation research is synthesis of a software solution for processing large volumes of text data and supporting the processing of Ukrainian and Russian texts.

Object of study: text data.

Subject of research: text data processing for Ukrainian and Russian languages with support for distributed computing.

Research Methods: In this dissertation, natural language processing methods based on rules, dictionaries and existing linguistic resources.

Scientific novelty: created software library for Apache Spark technology in, which, unlike the existing ones, contains sumarize functions and support for processing Ukrainian texts.

The practica lvalue of the obtained results is determined by the fact that the developed software library will automate the processing of text data.

Relationship with working with scientific programs, plans, topics: work was performed at the Department of Automated Information Processing and Management Systems of the National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute» within the topic «Natural text processings of tware library for Apache Spark». State Registration Number 0117U000924.

Testing: The main provisions of the work were reported and discussed at the conference "Informatics and Computer Engineering - IOT-2020".

Publications: Theses of the thesis are published in «Informatics and Computer Engineering - IOT-2020».

Keywords: NATURAL TEXT PROCESSING, LEMATIZATION, VECTORIZATION, SUMMARIZATION, TEXT STREAM.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

NLP (NaturalLanguageProcessing) – обробка природної мови

Ttokenization–токенізація, процес виділення з тексту окремих речень та слів.

Lematization - лематизація, процес переводу слова в початкову, словникову форму

TF-IDF - міра, що визначає важливість слова в контексті документу або корпусу документів

CBOW (Continuousbagofwords) – алгоритм виділення важливих слів з тексту

KEA (keyextractionautomatic) - алгоритм визначення ключових слів в тексті

ВСТУП

Бурхливе зростання кількості електронних документів, що спостерігається в даний час, наочно показує, що традиційні механізми обробки електронних документів не спроможні впоратись з потребами користувачів. Ця тенденція помітна як в мережі Інтернет, так і у корпоративних мережах. Таким чином, можна виділити основні проблеми, пов'язані зі збільшенням кількості інформації:

1. Більшість технологій роботи з текстовими документами орієнтовані на організацію зручної роботи з інформацією для людини, але практично відсутні можливості для передачі смислового змісту тексту, тобто відсутнє семантичне індексування.
2. Неструктурована інформація становить значну частину сучасних електронних текстових документів.

При сучасному темпі зростання обсягів інформаційних масивів, не важко уявити, якими надмірно трудомісткими процесами будуть, як класифікація всього фонду електронних текстових документів вручну, так і його кластеризація. Допомогти у вирішенні даної проблеми здатні програмні засоби, які автоматично виконують інтелектуальну обробку даних. Насамперед, мова йде про текстові документи, які можуть бути представлені у вигляді, придатному для автоматичного аналізу за допомогою програмних засобів. Не зважаючи на те, що Інтерес до даної проблеми, не тільки не згасає, але в останні два десятиліття є підвищеним - досі відсутнє рішення для української та російської мов.

Актуальність теми – відсутність інструментів для обробки великих об'ємів текстових даних, що підтримують українську та російську мови.

Мета – синтез програмного рішення для обробки великих об'ємів текстових даних та підтримкою обробки українськомовних та російськомовних текстів.

Призначення – обробка текстової інформації українською та російською мовами.

1 ТЕОРЕТИЧНІ ОСНОВИ

1.1. Попередняобробкатекстовихданих

Процес попередньої обробки вхідних даних є важливим та необхідним етапом в задачах обробки природної мови, на рівні з етапом самої класифікації чи розпізнання. Процес передобробки найчастіше використовується для виокремлення необхідної інформації з неструктурованих текстових даних[12]. Найчастіше тексти зашумлені зайвими даними, такі як дати, числа, слова, які не несуть семантичного змісту – прийменники, артиклі, займенники.

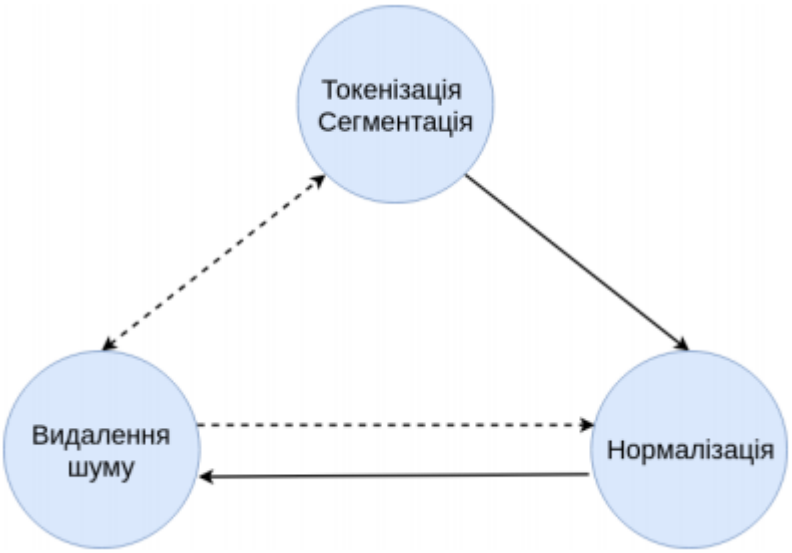


Рисунок 1.1 – Процес попередньої обробки тексту

Перший етап обробки називається токенізація (tokenization). Токенізація – це розбиття тексту на більш дрібні частини, токени. До токенів відносяться як слова, так і знаки пунктуації.

Наступний етап – нормалізація (normalization). Нормалізацією називають процес трансформації тексту, в єдину канонічну форму, в якій він міг не бути спочатку. Процес нормалізації вимагає розуміння того, який текст буде оброблятися далі, а який буде нормалізований. І в залежності від цього застосовують один або декілька з наведених нижче методів.

Зазвичай тексти містять різні граматичні форми одного і того ж слова, а також можуть зустрічатися однокореневі слова. Лематизації і стемінг мають на меті привести всі форми слова до однієї, нормальної словникової формі.

Стемінг – це грубий евристичний процес, який відрізає «зайве» від кореня і виділяє основну частину слова, часто це призводить до втрати словотворчих суфіксів так як основа не обов'язково збігається з морфологічним коренем слова.

Таблиця 1.1 – Приклад стемінгу

Слово	Стемінг
працюють	прац
швидкістю	швидк
ефективна	ефект

Лематизація – це більш тонкий процес, який використовує словник і морфологічний аналіз. В результаті лематизації від словоформи відкидаються флективні закінчення і повертається основна або словникова форма слова.

Таблиця 1.2 – Приклад леми

Слово	Лема
працюють	працювати
швидкістю	швидкість
ефективна	ефективний

Відмінність в тому, що стемер діє без знання контексту[17] і, відповідно, не розуміє різницю між словами, які мають різний зміст в залежності від частини мови. Однак у стемера є і свої переваги: його простіше впровадити і він працює

швидше. Також варто зауважити, що більш низька «точність» може не мати значення в деяких випадках.

Після нормалізації тексту його потрібно “очистити”, позбувшись стоп-слів.

Стоп-слова – це слова, які викидаються з тексту до/після обробки тексту. Коли ми застосовуємо машинне навчання до текстів, такі слова можуть додати багато шуму або створити аномалії, тому необхідно позбавлятися від нерелевантних слів. Під стоп-словами розуміють нецензурну лексику, вигуки, сполучники і т.д., які не несуть сенсу чи значеннях яких потрібно враховувати. Їх ще називають шумовими словами. При цьому потрібно розуміти, що не існує універсального списку стоп-слів, все залежить від конкретного випадку.

До типових стоп слів належать:

- цифри: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0;
- знаки пунктуації розташовані окремо: . , = + /! "; :%? * ();
- окремо розташовані букви алфавіту;
- службові частини мови;
- нецензурна мова.

Загалом, мета цих процедур – якнайбільше зменшення розмірності задачі без втрати емоційної складової.

1.2. Виділеннядодатковихознак з текстового документу

На другому етапі для кожного знайденого терму виділяються ознаки, за якими можна оцінити ступінь його важливості. Виділені ознаки можна розділити на 3 категорії: синтаксичні, статистичні, та структурні.

До синтаксичних ознак відносять ознаку частини мови, яка одержується за допомогою POS-розміток (part-of-speech tagging), та ознаки, отримані за допомогою різних онтологій та словників. Виявлення цих ознак є мовно залежним[2].

До статистичних ознак відноситься обчислення частотності слова у документах і в корпусі документів його схожості з іншими словами.

Застосування структурних ознак обумовлено тим, що важливість терміна для даного документа залежить від його місця розташування. Відзначають, що найчастіше ключові слова знаходяться в заголовку і першому параграфі тіла документа. В якості чисельної структурної ознаки можна використовувати нормалізовану відстань між словом і початком документу (відношення довжини ланцюжка слів від початку документа до поточного слова, поділене на кількість слів у документі)[3].

1.2.1. Міра TF IDF

Співвідношення **TF до IDF** – статистичний показник, який використовується переважно для оцінювання важливості конкретного слова(терміна) в контексті всього документа, що входить в загальну колекцію.

Термін TF IDF має англомовне походження, де TF означає частотність входження терміна, а IDF – зворотна (інвертована) частота документа.

TF або частота слова – це відношення кількості входження конкретного терміна до сумарного набору слів в досліджуваному тексті (документі). Цей показник відображає важливість (вагомість) слова в рамках певної статті / публікації.

$$tf(t,d) = \frac{n_t}{\sum_k n_k} \quad (1.1),$$

де n_t – число входжень слова t в документ, $\sum_k n_k$ – загальна кількість слів у документі.

IDF або зворотна частота документа – це інверсія частотності, з якої певне слово фігурує в колекції текстів (документів). Завдяки даним показникам можна знизити вагомість найбільш широко використовуваних слів (прийменників, сполучників, загальних термінів і понять).

Для кожного терміна в рамках певної бази текстів передбачається лише одне значення IDF.

$$idf(t, D) = log\frac{|D|}{\left|\{d_i \in D \mid t \in d_i\}\right|} \quad (1.2),$$

де $|D|$ – кількість документів у колекції, $\left|\{d_i \in D \mid t \in d_i\}\right|$ – кількість документів із колекції *D*, в яких зустрічається слово *t*.

Таким чином, міра TF-IDF є добутком двох множників:

$$tfidf(t, d, D) = tf(t,d) * idf(t, D) \quad (1.3)$$

Відповідно до TF IDF вагомість певного слова (терміна) прямо пропорційна до кількості разів його використання в конкретному тексті і обернено пропорційна від числа використання даного слова в інших документах.

Перевагами даної міри є:

- Швидкість обчислення. Для формування оцінки достатньо один раз просканувати документи в колекції.

- Врахування контексту документу. Розглядається не лише документ в якому знаходиться слово, але й усі документи колекції.

До недоліків можна віднести:

- Ігнорування контексту слова. Безглуздий документ, що перевантажений ключовими словами матиме високу оцінку.

- Частота слова ненадійний показник, особливо для української та російської мов. Наприклад текст, що містить багато синонімів релевантного слова матиме нижчу оцінку, ніж текст, перевантажений його омонімами.

1.2.2. Модель BagofWords

BagofWords – це подання тексту, що описує частоту слів у документі. Для реалізації моделі необхідно визначити дві речі:

- Словник відомих слів
- Міра присутності відомих слів.

Підхід називають «мішком» слів, оскільки будь-яка інформація про порядок чи структуру слів у документі відкидається. Модель враховує лише чи зустрічаються відомі слова в документі, а не де саме в документі вони знаходяться. На виході для кожного документу отримуємо словник де ключами є всі слова в корпусі документів, а значеннями скільки разів конкретне слово зустрічається у документі (Рисунок 1.2).

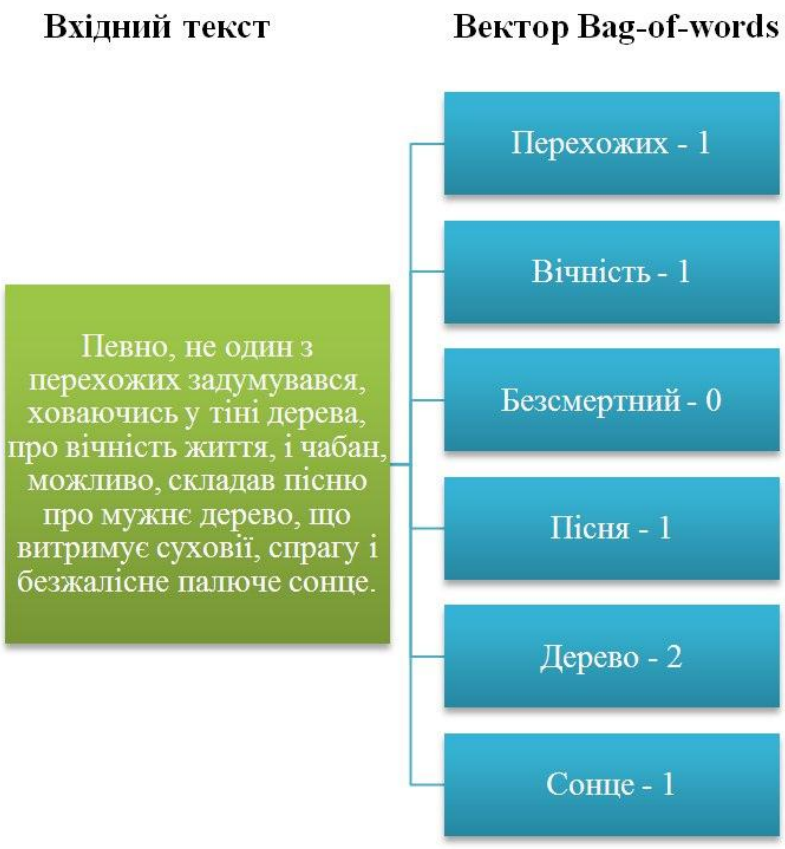


Рисунок 1.2 – Приклад роботи моделі Bag-of-words

Інтуїція полягає в тому, що документи подібні, якщо вони мають подібний набір слів. І, що лише із набору слів ми можемо дізнатися щось про значення документа.

Модель BugofWords дуже проста для розуміння та реалізації і пропонує велику гнучкість для налаштування конкретних текстових даних. Однак вона має досить значні недоліки:

- *Словник*: Словник вимагає ретельного проектування, зокрема для того, щоб керувати розміром, який безпосередньо впливає на розрідженість у представлені документу.
- *Розрідженість*: розріджені представлення складніші у моделюванні як з обчислювальних причин (складність простору та часу), так і з інформаційних, де проблема полягає в тому, що моделі використовують мало інформації у великому просторі представлень.
- *Значення*: відмова від порядку слів ігнорує контекст, а отже і значення слів у документі (семантику). Контекст і значення можуть значно покращити якість моделі. Зокрема це дає змогу правильно обробляти синоніми та порядок у якому написані слова (наприклад словосполучення “полювання на лисицю” та “лисиця на полюванні” складаються з однакового набору слів, але мають абсолютно різний сенс).

1.2.3. Векторна модель документу

Для подальшого аналізу необхідно побудувати векторну модель документа. Векторна модель – представлення колекції документів за допомогою векторів, що належать до одного, спільного для всієї колекції, векторного простору.

Векторна модель є основою для вирішення багатьох завдань інформаційного пошуку, таких як: пошук документа за запитом, класифікація документів, кластеризація документів.

Документ у векторній моделі розглядається як нерегульована множина термів. Для кожного терму присвоюється число – його важливість в корпусі

документів, визначена одним з методів наведених вище. Всі терми які зустрічаються в документі впорядковуються і з їх числових представлень будується вектор. Даний вектор є представленням документа у векторному просторі. Розмірність цього вектора, як і розмірність простору, дорівнює кількості різних термів у всій колекції, і є однаковою для всіх документів.

Формально вектор документа можна представити наступним чином:

$$d_j = (w_{1j}, w_{2j}, ..., w_{nj}) \quad (1.4),$$

де d_j – векторне представлення j-го документу,

w_{ij} – вага i-готерма в j-му документі,

n – загальна кількість різних термів в усіх документах колекції[4].

Маючи таке представлення для всіх документів, можна, наприклад, знаходити відстань між точками простору і тим самим вирішувати задачу подібності документів – чим ближче розташовані точки, тим більше схожі відповідні документи. У разі пошуку документа за запитом, запит теж представляється як вектор того ж простору – і можна обчислювати відповідність документів запиту.

1.2.4. Косинус подібності

Для подальшого аналізу необхідно визначити міру подібності векторів яка використовуватиметься для порівняння речень в в межах документу і документів в межах корпусу.

Загальновживаний підхід до знаходження подібних документів заснований на розрахунку максимальної кількості загальних слів між документами.

Але такий підхід має серйозні обмеження: у міру збільшення розміру документа, кількість спільних слів, як правило, збільшується, навіть якщо документи говорять на різні теми.

Косинус подібності допомагає подолати цей фундаментальний недолік у підході «підрахунку загальних слів» або евклідової дистанції.

Косинус подібності – це показник, який використовується для вимірювання схожості документів незалежно від їх розміру. Математично він вимірює косинус кута між двома векторами, що проектуються в багатовимірному просторі. Косинус подібності вигідно рахувати, оскільки навіть якщо два подібних документи знаходяться далеко один від одного на відстані Евкліда (через розмір документа), швидше за все, вони все одно можуть бути орієнтовані ближче. Чим менший кут, тим вище схожість косинуса (Рисунок 1.3).

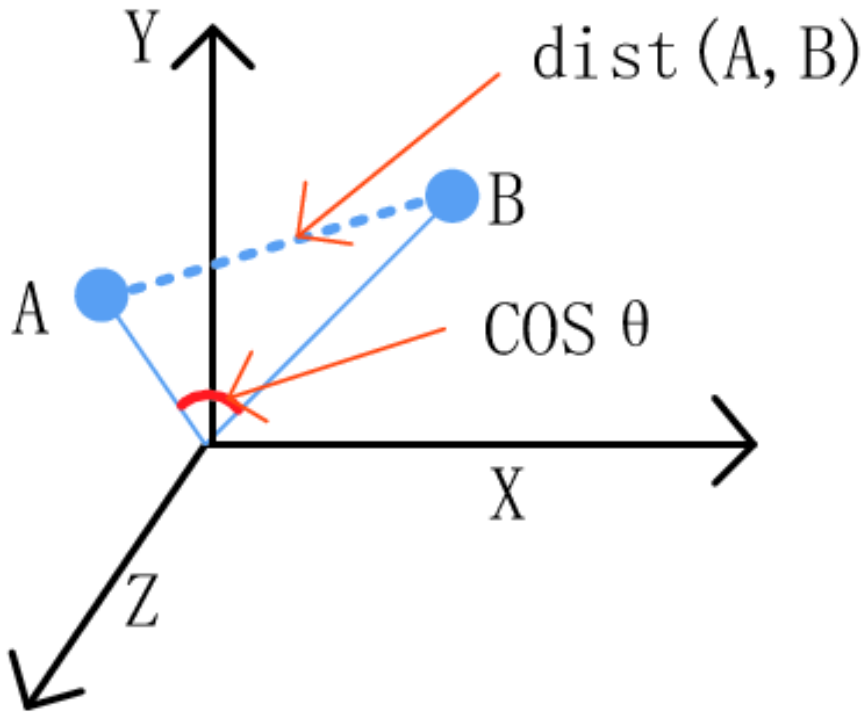


Рисунок 1.3 – Косинус подібності векторів

Варто зауважити, що коли корпус містить багато документів на різні теми важливо отримати якомога менший кут між документами на спільну тему і навпаки – великий кут, якщо теми різні.

У такому випадку потрібно враховувати семантичне значення термів. Тобто, слова, подібні за значенням, слід трактувати як подібні. Прикладом таких слів може бути “програмування” та “алгоритм”, “їжа” та “тарілка” та інші. В такому

випадку перетворення слів (термів) на відповідні вектори перед обрахуванням косинусу подібності може вирішити дану проблему.

1.2.5. Векторне представлення слів

Слова-вектори – це численні представлення слів, що зберігають семантичний зв'язок між ними. Наприклад, для слова “кішка” одним з найближчих буде слово “собака”. Однак векторне зображення слова олівець буде досить сильно відрізнятися від “кішки”. Ця схожість обумовлена частотою зустрічі цих двох слів (тобто [кішка, собака] або [кішка, олівець]) в одному контексті.

Цю модель запропонував у 2013 році чеський аспірант ТомашМиколов і назвав її **word2vec.** Запропонована модель дуже проста – необхідно передбачати ймовірність слова по його оточенню (контексту). Тобто необхідно підібрати такі вектори слів, щоб ймовірність, що привласнюється слову в даному контексті була близька до ймовірності зустріти це слово в цьому оточенні в реальному тексті написаному людиною.

Робота моделі здійснюється наступним чином: word2vec приймає великий корпус документів у якості вхідних даних і зіставляє кожному слову вектор, видаючи координати слів на виході. Спочатку модель генерує словник корпусу, а потім обчислює векторне подання слів, «навчаючись» на вхідних текстах. Векторне подання ґрунтується на контекстній близькості: слова, що зустрічаються в тексті поруч з однаковими словами (а отже, мають схожий зміст), будуть мати близькі (по косинусному відстані) вектори [5].

У word2vec реалізовані два основних алгоритму навчання: CBoW (англ. ContinuousBagofWords, «безперервний мішок зі словами") і Skip-gram.

CBoW (спрощена модель цієї архітектури описана вище) передбачає поточне слово, виходячи з навколишнього його контексту. Архітектура типу Skip-gram діє навпаки: вона використовує поточне слово, щоб передбачати навколишні його слова. Порядок слів контексту не впливає на результат в жодному з цих алгоритмів.

1.3. Реферування тексту

Реферування або підсумовування – це процес скорочення набору даних для створення підмножини (резюме), що представляє найважливішу або релевантну інформацію в межах оригінального вмісту.

Існує два загальні підходи до автоматичного реферування: вилучення (екстрактивний підхід) та абстрагування (абстрактивний підхід)[11].

При **вилученні** вміст витягується з вихідних даних, але витягнутий вміст жодним чином не змінюється. Приклади вилученого вмісту включають ключові фрази, які можна використовувати для "тегування" або індексації текстового документа, або ключових позицій (включаючи заголовки), які в сукупності містять абстрактні, і репрезентативні зображення або відео сегменти.

Методи даного підходу характеризують наявність оціночної функції важливості інформаційного блоку. Як правило, важливість речення визначається важливістю слів у ньому. Ранжуючи ці блоки за ступенем важливості і вибираючи необхідну їй кількість, ми формуємо підсумкове резюме тексту (Рисунок 1.4).

Рисунок 1.4 – Приклад екстрактивного реферування

Щодо тексту, вилучення є аналогом процесу скімінгу, де підсумок, заголовки та підзаголовки, цифри, перший та останній абзаци розділу читаються перед вибором щоб детально прочитати весь документ.

Абстрактні методи будують внутрішнє смислове подання оригінального тексту, а потім використовують це уявлення для створення реферату, ближчого до того, що може виразити людина. Якщо абстрактне застосування застосовується для узагальнення тексту в проблемах глибокого навчання, воно може подолати граматичні невідповідності екстрактивного методу. Алгоритми абстрактного узагальнення тексту створюють нові фрази та речення, які передають найкориснішу інформацію з оригінального тексту – як і люди(Рисунок 1.5).

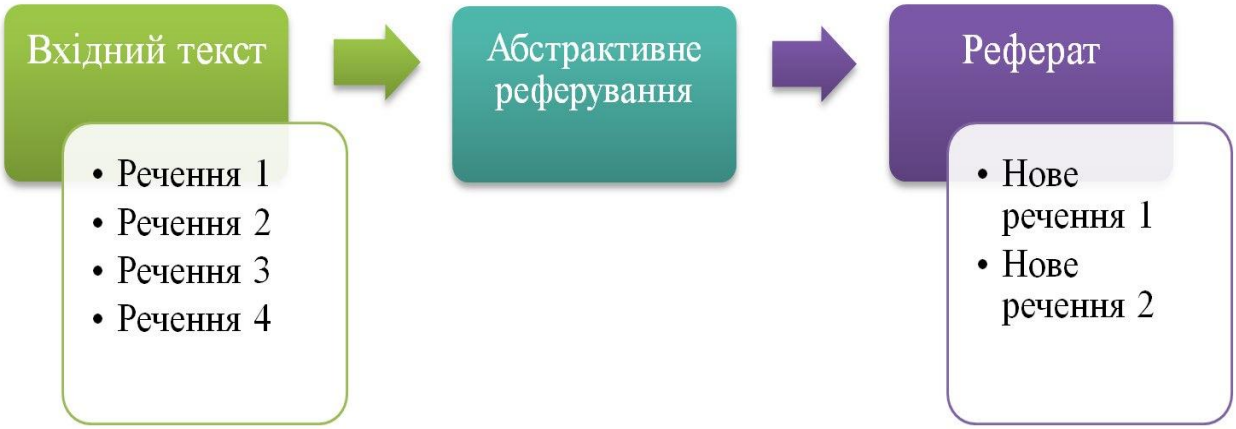


Рисунок 1.5 – Приклад абстрактивного реферування

Тому абстракція працює краще, ніж вилучення. Однак алгоритми узагальнення тексту, необхідні для абстрагування, надзвичайно складні у розробці; тому використання вилучення все ще популярне.

Основними методами вилучення є ранжування і відсікання. Зазвичай застосовують один з двох підходів – або використання будь-яких евристичних формул, які дозволяють визначити, чи є слово ключовим, або використання методів машинного навчання. Варто зазначити, що для машинного навчання з учителем необхідний попередньо розмічений корпус документів з виділеними ключовими словами. Спочатку застосування машинного навчання для виділення ключових термів зводилося до вирішення завдання бінарної класифікації шляхом

різних підходів до навчання класифікатора[6]. Використовувалися наївні байєсівські класифікатори, дерева прийняття рішень, бустінг. Однак такий підхід не дозволяє порівнювати знайдені терми між собою і вибирати кращі з них. Тому, згодом почали застосовуватися алгоритми, що дозволяють ранжувати терми попарно (наприклад, алгоритм KEA, TextRank)[7].

1.3.1. Латентний семантичний аналіз

Латентний семантичний аналіз – це алгебраїчно-статистичний метод, який виділяє приховані семантичні структури слів та речень. Це непідконтрольний підхід, який не потребує жодної підготовки чи додаткової інформації. LSA використовує контекст вхідного документа та витягує інформацію про те, які слова вживаються разом, а які загальні слова видно в різних реченнях. Велика кількість загальновживаних слів серед речень свідчить про те, що речення є семантично пов’язаними. Значення речення визначається словами, що містяться в ньому, а значення слів – з речень, які їх містять.

Алгоритми реферування, що базуються на латентному семантичному аналізі зазвичай містять 3 кроки: створення вхідної матриці, сингулярний розклад та вибір речень.

Створення вхідної матриці. Вхідний документ після попередньої обробки подають у вигляді матриці, де стовпці – це речення, а рядки – слова / фрази. Клітинки матриці використовуються для відображення важливості слів у реченнях. Для заповнення значень комірок можна використовувати різні підходи, основні з них були описані у розділі 1.2. Оскільки всі слова видно не у всіх реченнях, часто створена матриця є розрідженою. Спосіб створення вхідної матриці дуже важливий для узагальнення, оскільки він впливає на отримані матриці, обчислені за допомогою сингулярного розкладу. Це доволі повільний алгоритм, і його складність зростає із збільшенням розміру вхідної матриці, що погіршує продуктивність. Щоб зменшити розмір матриці, рядки матриці, тобто

слова, можна зменшити за допомогою їх попередньої обробки, яка описана в розділі 1.1.

Сингулярний розклад (з англ. SingularValueDecomposition), алгебраїчний метод, що використовується для з'ясування взаємозв'язків між реченнями та словами. Окрім можливості моделювання взаємозв'язків між словами та реченнями, SVD має можливість зменшення шуму, що допомагає підвищити точність.

У цьому методі дана вхідна матриця A розкладається на три нові матриці наступним чином:

$$A = U \sum v^T \quad (1.5),$$

де A – вхідна матриця(m x n), U – слова x вилученні поняття (m x n), $\sum$ – являє собою значення масштабування, v – речення x вилученні поняття (n x m).

Вибір речення: за допомогою результатів SVD використовуються різні алгоритми для вибору важливих речень. Серед них них варто виділити метод Гонга та Лю.

Алгоритм Гонга та Лю є одним з основних досліджень, проведених під час узагальнення тексту на основі LSA. Після представлення вхідного документа в матриці та обчислення значень SVD, матриця v^T використовується для вибору найважливіших речень. У матриці v^T порядок рядків вказує на важливість понять, так що перший рядок представляє найважливішу витягнуту концепцію. Значення комірок цієї матриці показують зв'язок між реченням і поняттям. Вище значення клітинки вказує на те, що речення більше пов'язане з поняттям.

У підході Гонга і Лю з найважливішого поняття вибирається одне речення, а потім друге речення з другого за значенням поняття, поки не буде зібрано заздалегідь визначену кількість речень.

LSA має кілька обмежень. Перший – це те, що він не використовує інформацію про порядок слів, синтаксичні відношення та морфології. Цей вид інформації може бути необхідним для з'ясування значення слів та текстів. Друге

обмеження полягає в тому, що воно використовує не світові знання, а лише інформацію, яка існує у вхідному документі. Третє обмеження пов'язане з роботою алгоритму. З більшими та неоднорідними даними продуктивність різко знижується. Зниження продуктивності спричинене SVD, який є дуже складним алгоритмом.

1.3.2. TextRank

TextRank – це алгоритм ранжування на основі графів. Він працює на основі PageRank алгоритму.

PageRank – сімейство алгоритмів оцінки важливості веб-сторінок за допомогою розв'язання систем лінійних рівнянь. Для кожної сторінки обчислює дійсне число, чим більше число – тим «важливіша» сторінка.

Замість прямого підрахунку кількості посилань PageRank інтерпретує посилання сторінки А на сторінку Б як голос сторінки А на користь сторінки Б. Після цього PageRank оцінює рейтинг сторінки відповідно до кількості отриманих голосів.

PageRank також враховує значимість кожної сторінки, що отримала голос, адже голоси деяких сторінок є важливішими, і відповідно до цього підвищується значущість сторінки, посилання на яку вони містять. Важливі сторінки отримують більш високу оцінку PageRank і відображаються на перших позиціях результатів пошуку. Для визначення значущості сторінки технологія Google використовує колективний інтелект всесвітньої мережі.

Розглянемо роботу PageRank на прикладі.

Припустимо, що існує мережа з чотирьох сторінок: A, B, C та D. Посилання сторінки саму на себе ігноруються. Всі посилання з одної сторінки на іншу обробляються, як одне посилання. PageRankініціалізується однаковими значеннями для всіх сторінок по 0.25 для кожної. PageRank, що переноситься з даної сторінки на цілі вихідних посилань на наступній ітерації розподіляється порівну між усіма вихідними посиланнями.

Припустимо, що сторінка B має посилання на сторінки C та A, сторінка C має посилання на сторінку A, а сторінка D має посилання на всі три сторінки. Таким чином, при першій ітерації, сторінка B буде переносити половину свого існуючого значення 0,125 на сторінку A, а іншу половину 0,125 на сторінку C. Сторінка C буде передавати все своє існуюче значення 0,25 до єдиної сторінки на яку він посилається A. Оскільки D має три вихідні посилання, вона передала би одну третину свого існуючого значення, або приблизно 0.083 на A. Після завершення цієї ітерації сторінка A матиме PageRank приблизно 0.458. Звідси можемо зробити висновок, що спрощена формула для обрахунку рангу сторінки U матиме наступний вигляд (рисунок 1.1).

$$PR(u) = \sum_{v \in B_u} sdfs\frac{PR(v)}{L(v)} \quad (1.6),$$

тобто значення PageRank для сторінки *u* залежить від значень PageRank для кожної сторінки *v*, що містяться в наборі B_u (наборі, що містить усі сторінки, що посилаються на сторінку *u*), розділеному на кількість *L(v)* посилань зі сторінки *v*[8].

TextRank можна використовувати як для реферування одного документа чи для реферації набору документів. Також слід зазначити, що схема для оцінки близькості вершин з використанням чисто статистичних величин, як TF-IDF має свої обмеження – наприклад, цей підхід зазвичай використовує лише частоту того, чи іншого слова, що відкидає певні семантичні особливості тексту. Звичайно, використання модифікованої функції для обчислення ваги ребра, яка буде враховувати синтаксичні та семантичні особливості тексту значно покращує результати реферату, але також звужує його застосування лише до однієї мови та зазвичай збільшує складність цього алгоритму.

1.4. Відомі програмні бібліотеки для обробки текстової інформації

На даний момент існує досить багато програмних засобів для обробки природної мови:

- NLTK (NaturalLanguageToolKit) – найвідоміша NLP бібліотека, створена дослідниками в цій галузі. Надає інструменти для символьної та статистичної обробки природних мов англійською мовою, написаних мовою програмування Python. В силу того, що бібліотека повністю написана на Python вона доволі повільна та не підходить для обробки великих наборів даних. Також недоліком є відсутність підтримки української та російської мов;

- SpaCy – це вдосконалена бібліотека для обробки природних мов на Python та Cython. Він побудований на найновіших дослідженнях і був розроблений з першого дня для використання в реальних продуктах. Він постачається з попередньо навченими статистичними моделями та векторами слів, і в даний час підтримує токенізацію для більше 49 мов. Українська та російська мови заявлені в списку підтримуваних мов, але наразі значна частина функціоналу для цих мов не реалізована. Також суттєвим мінусом є непристосованість бібліотеки до масштабування та розподілених обчислень;

- ApacheOpenNLP – це практична та доступна бібліотека з відкритим кодом. Написана на мові Java і використовує бібліотеки Java NLP з декораторами Python. Надає рішення для найпоширеніших завдань NLP: виявлення мови, токенізація, сегментація речень, позначення частини мови, виділення іменованих сутностей, шматування, синтаксичний аналіз.

- Бібліотеки CoreNLP, SparkNLP для технології ApacheSpark. Невід'ємним плюсом бібліотеки є повна підтримка бібліотеки Spark ML – однією з найпопулярніших на даний момент бібліотеки для машинного навчання технології ApacheSpark. Це означає легку масштабованість та підтримку розподілених обчислень. Проте бібліотека має і мінус попередників – відсутність підтримки для української мови (є лише підтримка російської мови). Існує підтримка мов програмування Python, Java та Scala.

Отже, виконавши аналіз відомих програмних засобів доходимо висновку, що не існує програмних бібліотек для одночасної обробки україномовних та російськомовних текстів з підтримкою високопродуктивних обчислень.

1.5. Постановка завдання

В рамках даної дисертаційної роботи мають бути вирішені наступні завдання:

- Аналіз існуючих методів та програмних засобів обробки текстової інформації.

- Розробка та проектування програмної бібліотеки обробки текстової інформації для програмної технології обробки надвеликих масивів інформації ApacheSpark підтримкою обробки українськомовних та російськомовних текстів.

- Дослідження ефективної розробленої бібліотеки шляхом виконання обчислювальних експериментів.

Для полегшення інтегрування в існуючі проекти, підвищення масштабування та швидкості роботи, програмне забезпечення необхідно оформити у вигляді бібліотеки та інтегрувати з фреймворкомApacheSpark.

Висновки до розділу

В розділі було досліджено методи обробки текстових даних та наведені відомості про основні етапи, які включає обробка. Детально розглянуто попередню обробку вхідного тексту та способи його векторизації з аналізом переваг та недоліків окремих підходів. Розглянуто підходи до реферування тексту та описано базові методи екстрактивного підходу.

2 ПІДХОДИ ДО РОЗПОДІЛЕНОЇ ОБРОБКИ ДАНИХ

Важливою особливістю програмного рішення, що планується розробити є розподілена обробка даних та масштабованість. Що дозволить використовувати його при обробці великих об'ємів текстових даних. В даному розділі буде проведена оцінка можливих інструментів та їх порівняння.

2.1. AmazonKinesis

AmazonKinesis спрощує збір, обробку та аналіз потокових даних в режимі реального часу, що дозволяє своєчасно отримувати інформацію і швидко реагувати на нову інформацію. AmazonKinesis може збирати і обробляти сотні гігабайт даних в секунду з сотень тисяч джерел, що дозволяє легко створювати додатки, що обробляють інформацію в режимі реального часу, з таких джерел, як потоки кліків на веб сайті, маркетинг і фінансова інформація, виробничі прилади та соціальні мережі, а також операційні журнали і дані вимірювань. Головним недоліком Kinesis[15] для виконання поставленої задачі, а саме обробки текстових потоків даних є важкість в виконані завдання. Ще одним недоліком Kinesis є його ціна в порівнянні з іншими продуктами, що і не дивно, враховуючи те що ми працюємо з AWS. Потрібно руками оптимізувати та налаштовувати його роботу, що зменшити ціну за використання, так наприклад можна агрегувати повідомлення розміром менше 25 кб в одне.

Приклад архітектури AmazonKinesis зображено на рисунку нижче (рисунок 2.1)

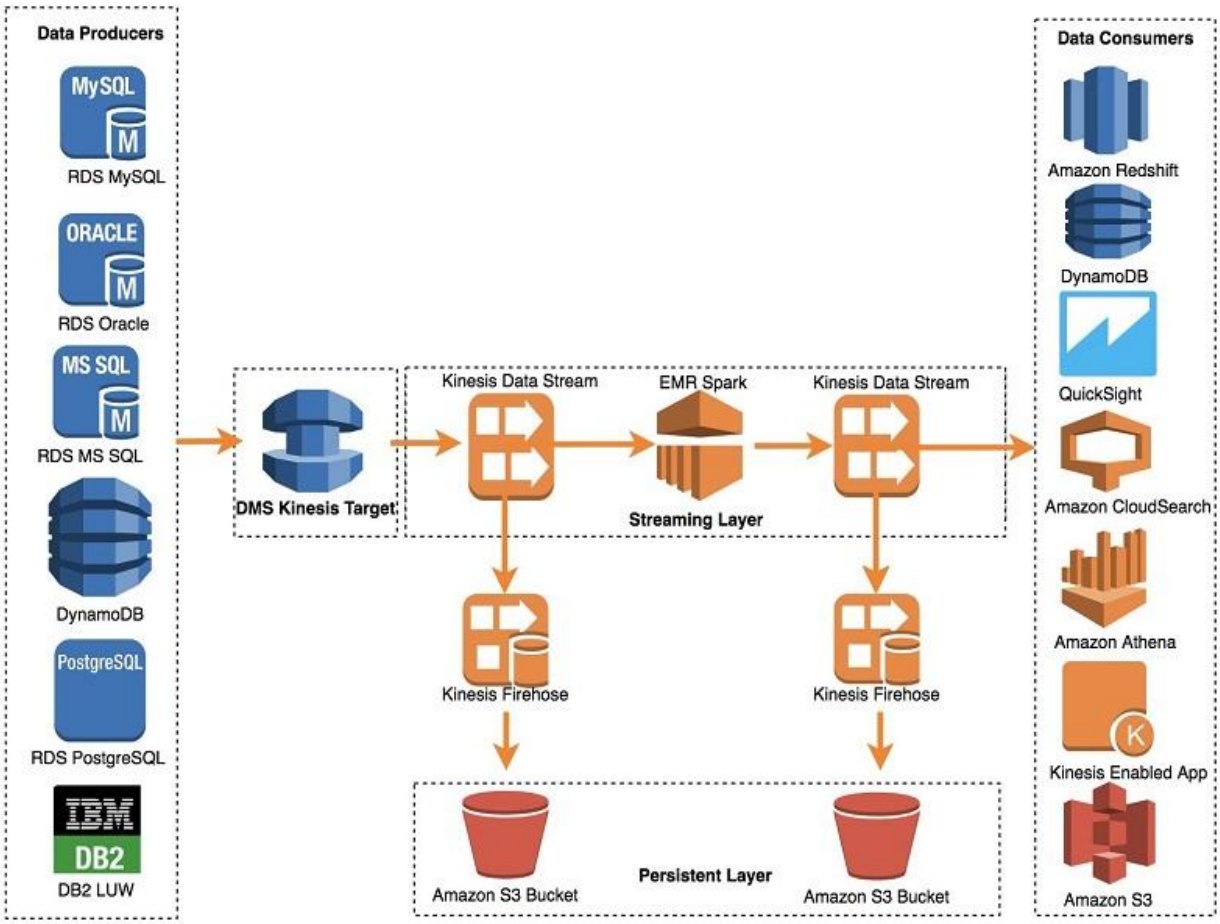


Рисунок 2.1 - Архітектура AmazonKinesis

Перевагами даного сервісу є:

- Масштабованість, що дозволяє обробляти дані з сотень тисяч джерел з низькими затримками.
- Робота в режимі реального часу, що дозволяє завантажувати і обробляти потокові дані в режимі реального часу, завдяки чому клієнт може отримувати аналітичну інформацію через секунди або хвилини замість годин або днів.

До недоліків сервісу належить:

- Належність до сервісів AWS. Це надає змогу легко інтегруватись з іншими сервісами amazon, але стає важко інтегруватися з сервісами, що не належать amazon.
- Доволі слабка документація, що ускладнює вивчення сервісу.

- Є проблеми з обробкою потоків. Інколи повідомлення з потоку дублюються.

2.2. ApacheHadoop

ApacheHadoop – це вільно поширюваний набір утиліт, бібліотек і фреймворк для розробки і виконання розподілених програм, що працюють на кластерах з сотень і тисяч вузлів. Ця основоположна технологія зберігання і обробки великих даних (BigData) є проектом верхнього рівня фонду ApacheSoftwareFoundation[14].

Коли ми говоримо про Hadoop, то в першу чергу маємо на увазі його файлову систему – HDFS. Найпростіший спосіб думати про HDFS – це уявити звичайну файлову систему, тільки більше. Звичайна ФС, за великим рахунком, складається з таблиці, файлових дескрипторів і області даних. У HDFS замість таблиці використовується спеціальний сервер – NameNode, а дані розкидані по серверам даних (DataNode).

Починаючи з версії 2.0. у Hadoop з’явився додатковий модуль **YARN** (англ. YetAnotherResourceNegotiator — «ще один ресурсний посередник»), що відповідає за управління ресурсами кластерів та планування задач.

Архітектура Hadoop зображена на рисунку нижче (рисунок 2.2).

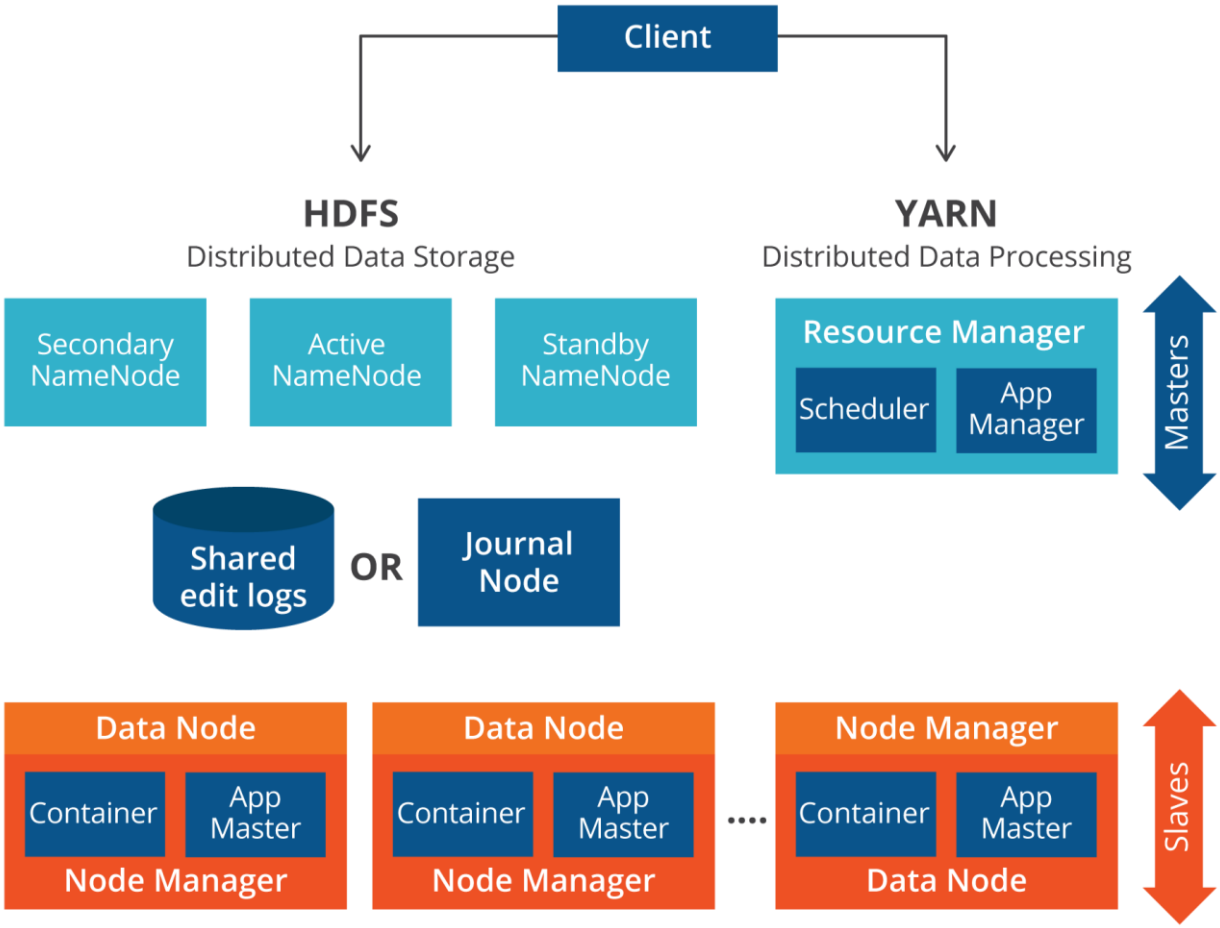


Рисунок 2.2. - Архітектура ApacheHadoop

Hadoopstreaming – це універсальний API, який використовується для роботи з потоковими даними. І картограф, і редуктор отримують свої вхідні дані у стандартному форматі. Вхідні дані беруться з stdin і висновок в stdout. Hadoop – це програма, яка використовується для обробки і зберігання великих даних.

Розробка Hadoop – це завдання обчислення великих даних з використанням різних мов програмування, таких як Java, Scala і інших.

2.3. ApacheSpark

ApacheSpark – це система обробки даних, яка може швидко виконувати завдання обробки на дуже великих наборах даних, а також може поширювати

завдання з обробки даних на декілька комп'ютерів, як самостійно, так і в тандемі з іншими розподіленими обчислювальними засобами. Ці дві якості є ключовими у світі великих даних та машинного навчання. Spark також знімає частину програмних тягарів цих завдань з плечей розробників за допомогою простого у користуванні програмного інтерфейсу, який абстрагує значну частину роботи розподілених обчислень та великої обробки даних[16].

Архітектура Spark базується на моделі абстракції еластичного розподіленого набору даних (RDD), яка є незмінною колекцією записів, розподілених на декількох комп'ютерах. Кожен RDD генерується з даних у зовнішніх надійних системах зберігання даних. Для забезпечення відмовостійкості інформація про трансформацію кожного RDD реєструється для побудови набору даних про рядки. Коли розділ даних RDD втрачається через збій вузла, RDD може перерахувати цей розділ з повною інформацією про те, як він був створений з інших RDD.

Spark має багато переваг:

- Простота у використанні. Spark надає користувачам більше 80 простих операторів високого рівня, які дозволяють користувачам розробляти паралельні застосунки на програмному рівні.
- Spark у 50 разів швидший ніж MapReduce в обробці пакетів.
- Загальна обчислювальна підтримка
- Гнучкий запуск. Spark може працювати в автономному режимі або ділитися кластером з іншими обчислювальними системами, такими як MapReduce.

Незважаючи на низку переваг, Spark також має слабкі сторони:

- Надмірне споживання ресурсів. Висока продуктивність досягається за рахунок розширення ресурсів зберігання.
- Проблеми з безпекою.
- Крива навчання. Від користувачів вимагається час, щоб вивчити модель та ознайомитися з наданим програмним інтерфейсом, перш ніж вони зможуть програмувати свої застосунки за допомогою Spark.

2.4. ApacheStorm

ApacheStorm (Сторм, Шторм) - це BigDataфреймворк з відкритим вихідним кодом для розподілених потокових обчислень в реальному часі, розроблений на мові програмування Clojure[13].

Storm легко інтегрується з уже використовуваними менеджерами черг і базами даних. Топологія Storm використовує потоки даних і обробляє їх як завгодно складними способами, перерозподіляючи потоки між етапами обчислень в міру необхідності.

Кластер ApacheStorm, що працює за принципом master-slave, складається з наступних компонентів:

- Ведучий вузол (master) із запущеною системною службою (демоном) Nimbus, який призначає завдання машинам і відстежує їх продуктивність;
- робочі вузли (workernodes), на кожному з яких запущений демон Supervisor, який визначає завдання іншим робочим вузлам та керує ними за необхідності.

Storm самостійно не відслідковує стан кластера, тому для зв'язку Nimbus із супервізорами та управлінням кластером використовується служба ZooKeeper. Архітектура ApacheStorm зображена на рисунку нижче (рисунок 2.3)

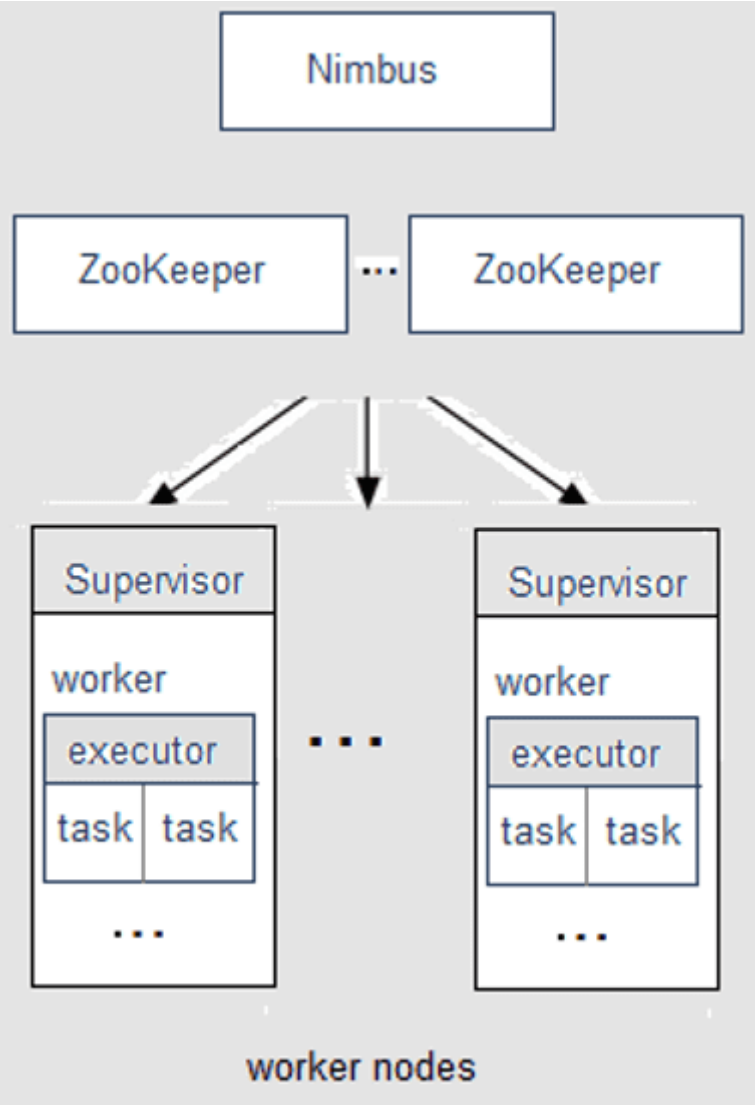


Рисунок 2.3 - архітектура ApacheStorm

До переваг ApacheStorm належить:

- інтеграція з будь-якими системами управління чергою та брокерами повідомлень (Kestrel, RabbitMQ, Kafka, JMS), а також базами даних;
- коли фонові завдання перестають працювати, Storm автоматично перезапустить їх на цьому ж вузлі кластера або на іншому, в разі його збою. Фреймворк самостійно обробляє параллелелізм, поділ даних і повтор дій в разі помилок;
- за рахунок інструменту розподіленого віддаленого виклику процедур Storm дозволяє виконувати кластерні обчислення на вимогу, коли клієнт синхронно очікує результат;

- Storm забезпечує обробку потокових даних в реальному часі з затримкою менше 1 секунди, показуючи кращі результати в порівнянні з ApacheSpark.

До недоліків ApacheStorm належить:

- відсутність можливостей гнучкої обробки подій в різних періодах, наприклад, часові та сеансові вікна, як в ApacheKafkaStreams і Flink;
- не підтримує управління станом додатків (stateful)

Висновки за розділом

В даному розділі було порівняно наявні програмні засоби та системи для потокової обробки даних, наведено їх недоліки та переваги.

Серед запропонованих варіантів було обрано програмний комплекс ApacheSpark. Він має всі необхідні функції для виконання поставленої задачі, а саме обробку в оперативній пам’яті, гнучкість в масштабуванні та налаштуванні системи. За замовчуванням є вся необхідна інфраструктура та можливість інтеграції з іншими системами які будуть використані в процесі дослідження, ElasticSearch та Kafka. Також важливою можливістю Spark є підтримка віконної обробки, що дозволить зменшити час на векторизацію документів. Тому ApacheSpark це найкращий варіант для виконання дослідження.

3 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Аналізвимог

Для кращого формулювання вимог розділимо їх на функціональні та нефункціональні. До перших віднесемо ті, що стосуються конкретно набору функцій, а саме: формат вводу та виводу даних, можливості для обробки. Тобто алгоритмічна складова системи. До інших віднесемо бізнес складову системи.

До функціональних вимог належать:

- можливість прийому текстових даних у вигляді файла, рядка, RDD, потоку даних;
- визначення мови кожного окремого тексту;
- отримання інформації про кожне слово в тексті, як про частину мови;
- векторизація тексту;
- отримання числової оцінки подібності двох текстів;
- отримання реферату по тексту з заданим рівнем точності.

До нефункціональних вимог належать:

- програмне забезпечення повинне бути оформлене у вигляді бібліотеки.

3.2. Менеджер пакетів

Оскільки дана автоматизації обробки текстів виникає в багатьох сферах: промислових і не тільки, головною задачею є максимальне полегшення повторного використання коду. Саме тому дане програмне рішення було оформлено у вигляді програмної бібліотеки. Завдяки цьому для початку його інтеграції в довільний проект досить виконати лише одну команду *pipinstall* а всю роботу по завантаженню вихідного коду, встановленню залежностей, перевірку версій та розміщенню їх у проекті виконує менеджер пакетів мови програмування Python - pip.

Пакетні менеджери спрощують використання чужого коду, надаючи цей код у вигляді незалежних модулів - пакетів. Ці пакети підключаються до власного

коду за принципом чорних ящиків - користувач не знає і йому не важливо як все влаштовано всередині цього ящика, але знає, що він робить. Завдяки такій слабосвязанной архітектурі з'являється можливість легко оновлювати чужий код або замінювати один пакет іншим зі схожою функціональністю.

У кожного пакетного менеджера є файл з налаштуваннями, в якому необхідно вказати від яких пакетів залежить код, щоб пакетний менеджер їх скачав і встановив в систему. При цьому кожен пакет може залежати від інших пакетів. Пакетний менеджер розплутує цю систему залежностей і встановлює все що потрібно, тому їх ще називають менеджерами залежностей.

Наприклад програмна бібліотека spark_lp залежить від pymorphy2[10]. При встановлені бібліотеки в проект пакетний менеджер це побачить та перевірить, чи наявна вона в проекті та чи виконується вимога щодо версії, якщо ні - встановить її або оновить до необхідної версії.

З ростом кодової бази використання менеджера залежностей стає необхідною складовою в роботі. Тому pip було включено до інсталятора Python з версій 3.4 для Python 3 та 2.7.9 для Python 2, і він використовується багатьма проектами Python.

3.3. Архітектурапрограмногозабезпечення

Бібліотека може бути умовно розділена на дві частини:

- обробка окремого документу
- обробка корпусу або потоку документів

Обробка окремого документу виконується в класах Text, TextRDD. Вони ідентичні за інтерфейсом проте використовують різні абстракції при обробці даних.

TextRDD працює з типом даних Spark RDD.

RDD (ResilientDistributedDataset) - це основна структура даних Spark. Це незмінна розподілена колекція об’єктів. Кожен набір даних у RDD розділений на логічні розділи, які можна обчислити на різних вузлах кластера. RDD можуть

містити будь-який тип об’єктів Python, Java або Scala, включаючи визначені користувачем класи.

Формально RDD - це лише для читання розділена колекція записів. RDD можна створити за допомогою детермінованих операцій над даними на стабільному сховищі або іншими RDD. RDD - це відмовостійка колекція елементів, яка може експлуатуватися паралельно.

Існує два способи створення RDD - паралелізація існуючої колекції у вашій програмі драйвера або посилання на набір даних у зовнішній системі зберігання, наприклад, спільна файлова система, HDFS, HBase або будь-яке джерело даних, що пропонує вхідний формат Hadoop.

Spark використовує концепцію RDD для досягнення більш швидких та ефективних операцій MapReduce.

Клас Text повністю повторює функціонал TextRDD, але на відміну від останнього базується на вбудованих типах даних мови Python, що дозволяє використовувати його при роботі з корпусом документів на Spark.

Вхідні дані:

- Необроблений текст у вигляді рядка

Вихідні дані:

- мова тексту
- кількість слів в тексті
- структура тексту: окремі речення і слова
- інформація про кожне слово: нормальна форма, частина мови, рід, число, відмінок.
- відфільтровані речення (очищені від стоп-слів)
- міра TF-IDF для кожного слова в контексті даного тексту
- реферат тексту

Робота з корпусом документів виконується за допомогою класів TextCorpus та TextStream. Як і у випадку з Text та TextRDD дані класи ідентичні по інтерфейсу, проте відрізняються способом обробки даних.

TextsCorpus приймає на вхід список обо RDD необроблених документів, та обробляє їх на етапі ініціалізації за допомогою класу Text. Після попередньої структуризації та токенізації бібліотека приводить документи до векторного представлення шляхом обчислення міри TF-IDF. На цьому етапі користувач має змогу отримати TF-IDF будь-якого документу, що входить в корпус, та порахувати косинус подібності між парою документів.

Клас TextsStream приймає на вхід об'єкт SparkStream та проводить аналогічні операції, використовуючи при цьому функцію SlidingWindow.

SlidingWindow - це особливість SparkStreaming, що забезпечує віконні обчислення, які дозволяють застосовувати перетворення над “ковзаючим” вікном даних. На рисунку нижче показано це вікно (рисунок 3.1.).

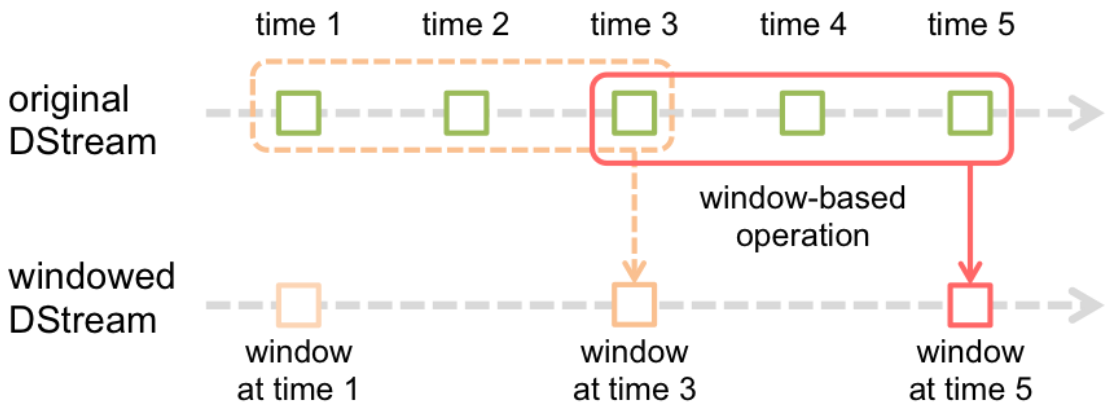


Рисунок 3.1. - Ковзаюче вікно

Вхідні та вихідні дані при обробці корпусу документів також дещо відрізняються.

Вхідні дані:

- Список необроблених документів

Вихідні дані:

- список оброблених документів
- міра TF-IDF для кожного слова в контексті корпусу документів

- векторне представлення документу
- косинус подібності кожної пари документів

Загальну архітектуру проекту, який користуватиметься даною бібліотекою[9] можна представити у вигляді наступної схеми (рисунок 3.3)

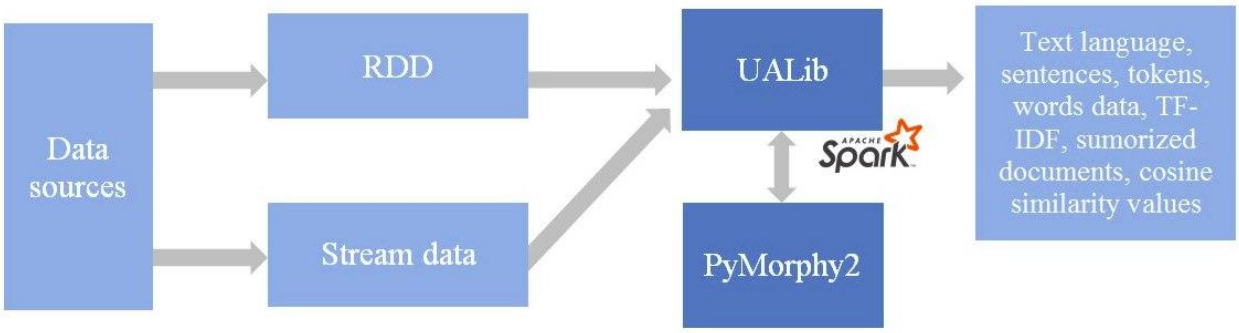


Рисунок 3.3 - Архітектура проекту

3.4. Інтерфейсбібліотеки

Таблиця 3.1. Опис інтерфейсу бібліотеки

Назва функції	*Аргументи*	*Значення, яке повертається*	*Призначення*
clean_text	text: String - неочищенний текст	text: String - очищений текст	Очищення тексту від зайвих символів на кшталт табуляції та переносу рядка. Видалення повторюваних символів.

Продовження таблиці

Назва функції	*Аргументи*	*Значення, яке повертається*	*Призначення*
split_to_sentences	text: String - очищений текст	sentences: List - список речень	Розбиває текст на масив речень.
split_to_words	sentence: String - речення	words: List - список слів з заданого речення	Розбиває речення на масив слів.
normalize_word	word: String - ненормалізоване слово, lang: {UK, RU} - мова	word: String - слово у нормальній формі	Переводить слово у початкову форму за допомогою зовнішньої бібліотеки pymorphy2
normalize_sent	sentence: List - речення, у вигляді списку слів, lang: {UK, RU} - мова	sentence: List - список нормалізованих слів	Переводить кожне слово в реченні у нормальну форму, використовуючи функцію normalize_word

Продовження таблиці

Назва функції	*Аргументи*	*Значення, яке повертається*	*Призначення*
parse_sent	sentence: List - речення, у вигляді списку слів, lang: {UK, RU} - мова	tokens: List - список об'єктів Parse, які містять інформацію про слова у реченні	Парсить кожне слово у даному реченні та повертає інформацію про нього (число, рід, частина мови)
parse_obj_to_dict	parse_obj: Object - обєкт бібліотеки pymorphy2, який містить інформацію про слово	parse_dict: Dictionary - інформація про слово у вигляді словника ключ: значення	Переводить інформацію про слово у формат ключ: значення
normalize_text	text: List - структурований текст у вигляді списку речень і слів, lang: {UK, RU} - мова	text: List - структурований та нормалізований текст	Нормалізує кожне слово у тексті

Продовження таблиці

Назва функції	*Аргументи*	*Значення, яке повертається*	*Призначення*
filter_stop_words	text: List - структурований текст, lang: {UK, RU} - мова, stop_words: Set - множина стоп слів	text: List - відфільтрований текст	Фільтрує текст від стоп-слів. В якості стоп слів виступає переданий список або збережений список частовживаних слів для заданої мови
get_stop_words	lang: {UK, RU} - мова	stop_words: Set - множина стоп слів	Для заданої мови повертає найбільш вживані слова, які не несуть за собою інформаційної цінності
update_vectors	v1: Vector, v2: Vector - вектори	v1: List, v2: List - оновлені вектори у вигляді списків	Оновлює вектори таким, чином, щоб вони мали однакову розмірність
cos_sim	v1: Vector, v2: Vector - вектори	similarity: Float - косинус подібності	Обробляє вектори та рахує їх косинус подібності
get_tfidf	text: String	tfidf: Vector - вектор tfidf	Повертає векторне представлення документу у вигляді TF IDF міри слів цього документу в контексті корпусу

Продовження таблиці

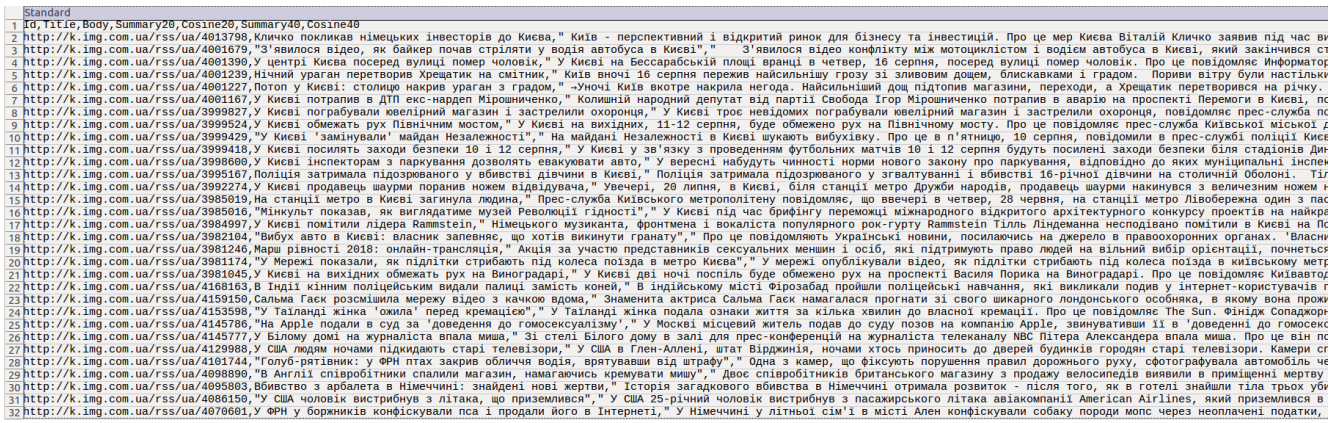
Назва функції	*Аргументи*	*Значення, яке повертається*	*Призначення*
get_similarity	text1: String, text2: String - невідформатовані документи	similarity: Float - косинус подібності	Нормалізує два документи та переводить їх у векторне представлення за допомогою TF IDF міри. Після чого рахує косинус подібності між отриманими векторами.
sumarize	text: String - текст	summary: String	Реферує заданий текст.

3.5. Прикладвикористаннябібліотеки

В якості додатку для демо-показу був розроблений комплекс, що складається з 4 сервісів: постачальник, сервіс транспортування, обробник, сховище даних та візуалізатор.

3.5.1. Постачальник повідомлень

В якості постачальника виступає невеликий сервіс, реалізований на мові програмування Python. Головним завданням сервісу є імітація потоку даних. Джерелом даних слугує завчасно збережений файл з 1000 текстів новин (рисунок 3.4).



Id,title,Body,Summary20,Cosine20,Summary40,Cosine40
http://k.img.com.ua/rss/ua/4013708,Кличко покликав німецьких інвесторів до Києва," Київ - перспективний і відкритий ринок для бізнесу та інвестицій. Про це мер Києва Віталій Кличко заявив під час ви
http://k.img.com.ua/rss/ua/4001679,"З'явилося відео, як байкер почав стріляти у водія автобуса в Києві"," З'явилося відео конфлікту між мотоциклістом і водієм автобуса в Києві, який закінчився ст
http://k.img.com.ua/rss/ua/4001390,У центрі Києва посеред вулиці помер чоловік," У Києві на Бессарабській площі вранці в четвер, 16 серпня, посеред вулиці помер чоловік. Про це повідомляє Інформатор
http://k.img.com.ua/rss/ua/4001239,Нічний ураган перетворив Хрещатик на смітник," Київ вночі 16 серпня пережив найсильнішу грозу зі зливовим дощем, блискавками і градом. Пориви вітру були настільки
http://k.img.com.ua/rss/ua/4001227,Потоп у Києві: столицю накрив ураган з градом," →Уночі Київ вкотре накрила негода. Найсильніший дощ підтопив магазини, переходи, а Хрещатик перетворився на річку.
http://k.img.com.ua/rss/ua/4001167,У Києві потрапив в ДТП екс-нардеп Мірошниченко," Колишній народний депутат від партії Свобода Ігор Мірошниченко потрапив в аварію на проспекті Перемоги в Києві, по
http://k.img.com.ua/rss/ua/3999827,У Києві пограбували ювелірний магазин і застрелили охоронця," У Києві троє невідомих пограбували ювелірний магазин і застрелили охоронця, повідомляє прес-служба по
http://k.img.com.ua/rss/ua/3999524,У Києві обмежать рух Північним мостом," У Києві на вихідних, 11-12 серпня, буде обмежено рух на Північному мосту. Про це повідомляє прес-служба Київської міської д
http://k.img.com.ua/rss/ua/3999429,"У Києві 'замінували' майдан Незалежності"," На майдані Незалежності в Києві шукають вибухівку. Про це в п'ятницю, 10 серпня, повідомили в прес-службі поліції Киє
http://k.img.com.ua/rss/ua/3999418,У Києві посилять заходи безпеки 10 і 12 серпня," У Києві у зв'язку з проведенням футбольних матчів 10 і 12 серпня будуть посилені заходи безпеки біля стадіонів Дин
http://k.img.com.ua/rss/ua/3998600,У Києві інспекторам з паркування дозволять евакуювати авто," У вересні набудуть чинності норми нового закону про паркування, відповідно до яких муніципальні інспек
http://k.img.com.ua/rss/ua/3995167,Поліція затримала підозрюваного у вбивстві дівчини в Києві," Поліція затримала підозрюваного у згвалтуванні і вбивстві 16-річної дівчини на столичній Оболоні. Тіл
http://k.img.com.ua/rss/ua/3992274,У Києві продавець шаурми поранив ножем відвідувача," Увечері, 20 липня, в Києві, біля станції метро Дружби народів, продавець шаурми накинувся з величезним ножем н
http://k.img.com.ua/rss/ua/3985019,На станції метро в Києві загинула людина," Прес-служба Київського метрополітену повідомляє, що ввечері в четвер, 28 червня, на станції метро Лівобережна один з пас
http://k.img.com.ua/rss/ua/3985016,"Мінкульт показав, як виглядатиме музей Революції гідності"," У Києві під час брифінгу переможці міжнародного відкритого архітектурного конкурсу проектів на найкра
http://k.img.com.ua/rss/ua/3984997,У Києві помітили лідера Rammstein," Німецького музиканта, фронтмена і вокаліста популярного рок-гурту Rammstein Тілль Ліндеманна несподівано помітили в Києві на По
http://k.img.com.ua/rss/ua/3982104,"Вибух авто в Києві: власник запевняє, що хотів викинути гранату"," Про це повідомляють Українські новини, посилаючись на джерело в правоохоронних органах. 'Власни
http://k.img.com.ua/rss/ua/3981246,Марш рівності 2018: онлайн-трансляція," Акція за участю представників сексуальних меншин і осіб, які підтримують право людей на вільний вибір орієнтації, почнеться
http://k.img.com.ua/rss/ua/3981174,"У Мережі показали, як підлітки стрибають під колеса поїзда в метро Києва"," У мережі опублікували відео, як підлітки стрибають під колеса поїзда в київському метр
http://k.img.com.ua/rss/ua/3981045,У Києві на вихідних обмежать рух на Виноградарі," У Києві дві ночі поспіль буде обмежено рух на проспекті Василя Порика на Виноградарі. Про це повідомляє Київавтод
http://k.img.com.ua/rss/ua/4168163,В Індії кінним поліцейським видали палиці замість коней," В індійському місті Фірозабад пройшли поліцейські навчання, які викликали подив у інтернет-користувачів г
http://k.img.com.ua/rss/ua/4159150,Сальма Гаєк розсмішила мережу відео з качкою вдома," Знаменита актриса Сальма Гаєк намагалася прогнати зі свого шикарного лондонського особняка, в якому вона прожи
http://k.img.com.ua/rss/ua/4153598,"У Таїланді жінка 'ожила' перед кремацією"," У Таїланді жінка подала ознаки життя за кілька хвилин до власної кремації. Про це повідомляє The Sun. Фінідж Сопаджорн
http://k.img.com.ua/rss/ua/4145786,"На Apple подали в суд за 'доведення до гомосексуалізму'"," У Москві місцевий житель подав до суду позов на компанію Apple, звинувативши її в 'доведенні до гомосекс
http://k.img.com.ua/rss/ua/4145777,У Білому домі на журналіста впала миша," Зі стелі Білого дому в залі для прес-конференцій на журналіста телеканалу NBC Пітера Александера впала миша. Про це він по
http://k.img.com.ua/rss/ua/4129988,У США людям ночами підкидають старі телевізори," У США в Глен-Аллені, штат Вірджинія, ночами хтось приносить до дверей будинків городян старі телевізори. Камери сп
http://k.img.com.ua/rss/ua/4101744,"Голуб-рятівник: у ФРН птах закрив обличчя водія, врятувавши від штрафу"," Одна з камер, що фіксують порушення правил дорожнього руху, сфотографувала автомобіль че
http://k.img.com.ua/rss/ua/4098890,"В Англії співробітники спалили магазин, намагаючись кремувати мишу"," Двоє співробітників британського магазину з продажу велосипедів виявили в приміщенні мертву
http://k.img.com.ua/rss/ua/4095803,Вбивство з арбалета в Німеччині: знайдені нові жертви," Історія загадкового вбивства в Німеччині отримала розвиток - після того, як в готелі знайшли тіла трьох уб
http://k.img.com.ua/rss/ua/4086150,"У США чоловік вистрибнув з літака, що приземлився"," У США 25-річний чоловік вистрибнув з пасажирського літака авіакомпанії American Airlines, який приземлився в
http://k.img.com.ua/rss/ua/4070601,У ФРН у боржників конфіскували пса і продали його в Інтернеті," У Німеччині у літньої сім'ї в місті Ален конфіскували собаку породи мопс через неоплачені податки,

Рисунок 3.4. - Джерело даних

Кожні n секунд з набору обирається випадковий текст та передається в сервіс транспортування - топік kafka. Цей підхід дозволяє імітувати контрольований потік даних, та регулювати необхідну щільність повідомлень та навантаження.

3.5.2. Сервіс транспортування повідомлень

В якості сервісу транспортування було обрано ApacheKafka.

ApacheKafka - менеджер обміну повідомленнями на платформі Java. У Kafka працює по принципу publish/subscribe, тобто є тема повідомлень, до якої видавці пишуть повідомлення і є підписники на кожну тему, які читають ці повідомлення. Всі повідомлення в процесі доставки записуються на диск і не залежать від підписників (рисунок 3.5).

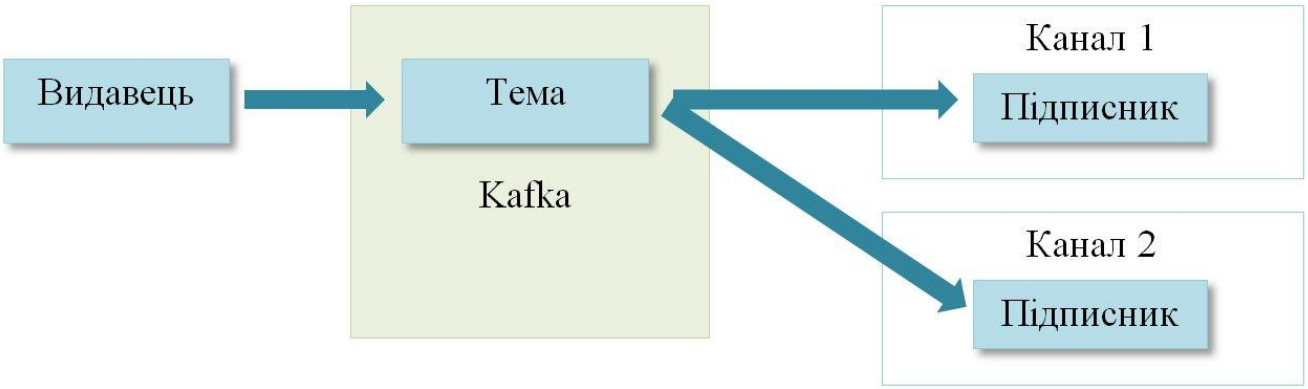


Рисунок 3.5 - Архітектура системи з Kafka

Крім того, для полегшення моніторингу цієї підсистеми вона включає дві служби, kafka-reverse-proxy і kafka-ui. Перша необхідна для отримання доступу до

черги через протокол http / https або обгортання існуючої черги повідомлень.

Kafka-ui - це веб-інтерфейс, який можна відкрити в браузері. Він має всі інструменти, необхідні для моніторингу, маніпуляції та управління чергами повідомлень. Приклад інтерфейсу наведений на рисунку 3.6.

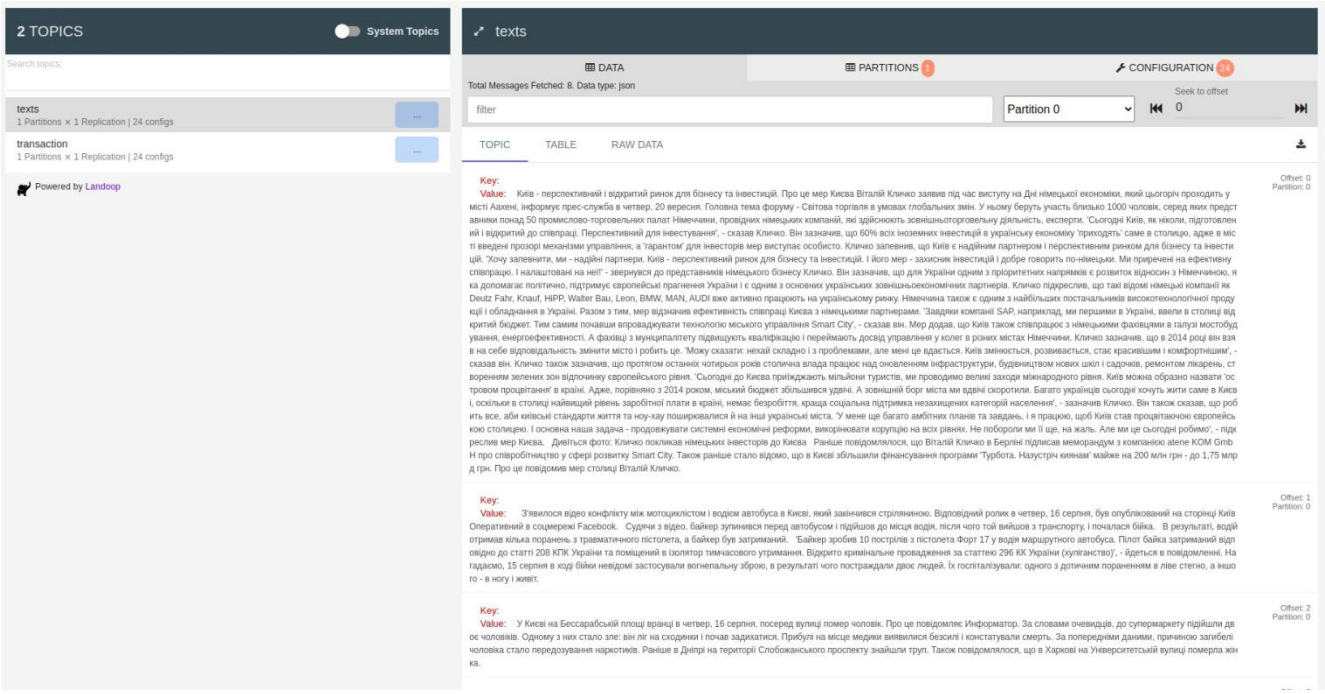


Рисунок 3.6 - Веб інтерфейс kafka-ui

Також для управління всім станом підсистеми, збереження конфігурації про черги, топіки та повідомлення використовується сервіс zoo-keeper.

ZooKeeper - це централізована служба для збереження інформації про конфігурацію, іменування, забезпечення розподіленої синхронізації та надання групових послуг. Всі ці види послуг використовуються в тій чи іншій формі розподіленими програмами.

3.5.3. Сервіс обробник

В якості сервісу обробника виступає скрипт написаний на мові програмування Python. Скрипт отримує потік текстових даних з ApacheKafka та використовує розроблену бібліотеку, щоб розбити даний потік на вікна та обробити його. Результатом обробки є структурована інформація про токени-слова в тексті, загальну

кількість слів та, розраховану в контексті вікна, міру TF-IDF. Отримані дані записуються в сховище даних Elasticsearch.

Elasticsearch - це високо масштабований механізм повнотекстового пошуку та аналітики з відкритим кодом. Він дозволяє зберігати, шукати та аналізувати великі обсяги даних швидко та майже в реальному часі. Elasticsearch забезпечує розподілену систему на вершині LuceneStandardAnalyzer для індексації та автоматичного вгадування типу та використовує REST API на основі JSON для посилання на функції Lucene.

3.5.4. Візуалізація

Після запису даних у нереляційну базу даних elastic, вони автоматично потрапляють до служби візуалізації - Kibana. Kibana має функцію автоматичного оновлення даних з індексу кожні n секунд, що допомагає при перегляді потоку та дає можливість моніторити дані в реальному часі.

Kibana - це сервіс візуалізації даних Elasticsearch та навігаційна служба ElasticStack. Вона допомагає створювати інформаційні панелі, налаштовувати форму візуалізації, створювати інтерактивні графіки, навіть представляти геодані, аналізувати взаємозв'язки та вивчати аномалії за допомогою машинного навчання. В якості прикладу було створено таблицю, що записує дані у форматі

- текст;
- кількість слів;
- середня міра tf-idf.

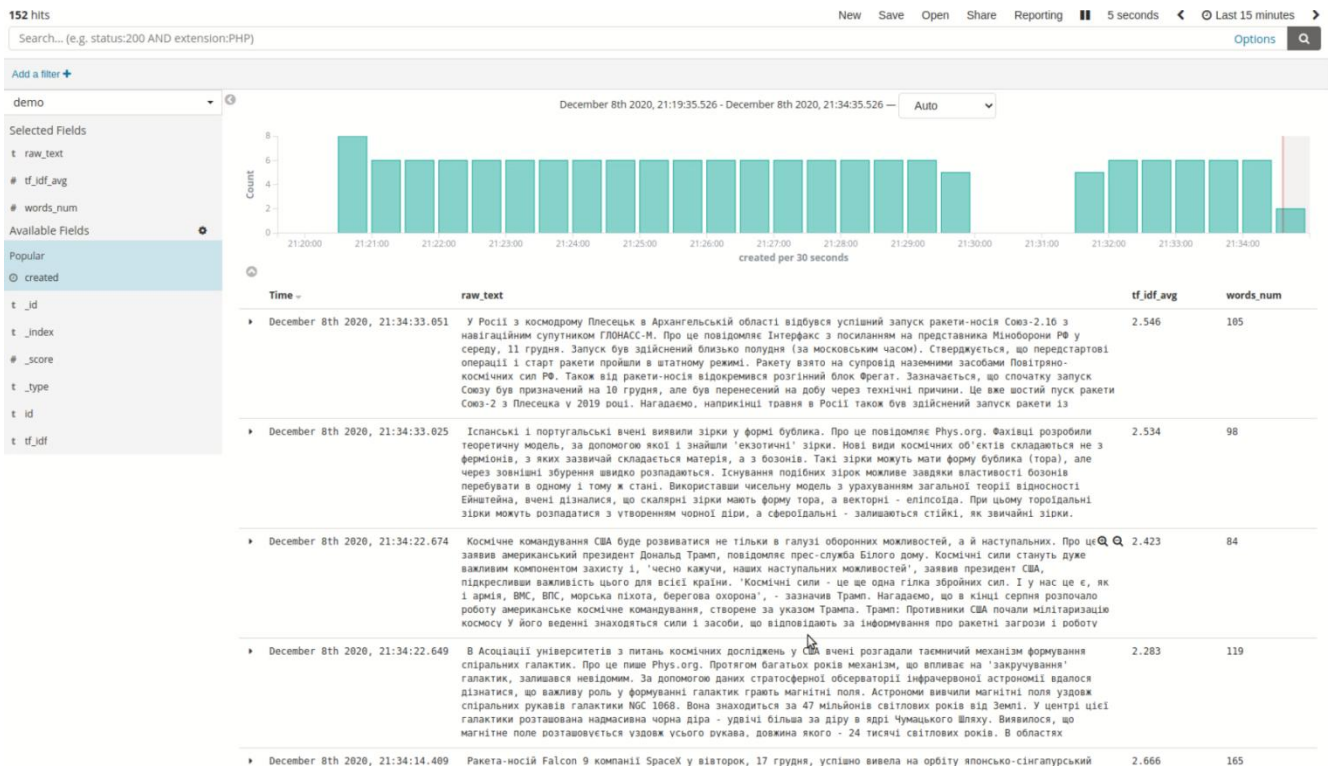


Рисунок 3.6 - Приклад таблиці в Kibana

4 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ

4.1. Апаратнезабезпечення

Для тестування використовувався персональний комп'ютер з наступними характеристиками:

- Операційна система: Ubuntu 16.04
- Процесор: 8 ядерний CPU: Intel(R) Core(TM) i7-8500 CPU @ 4.00GHz;
- Відеокарта: AMD Radeon 4 Gb
- RAM: 16Gb Kingston DDR3 SDRAM

4.2. Швидкістьобробкитексту

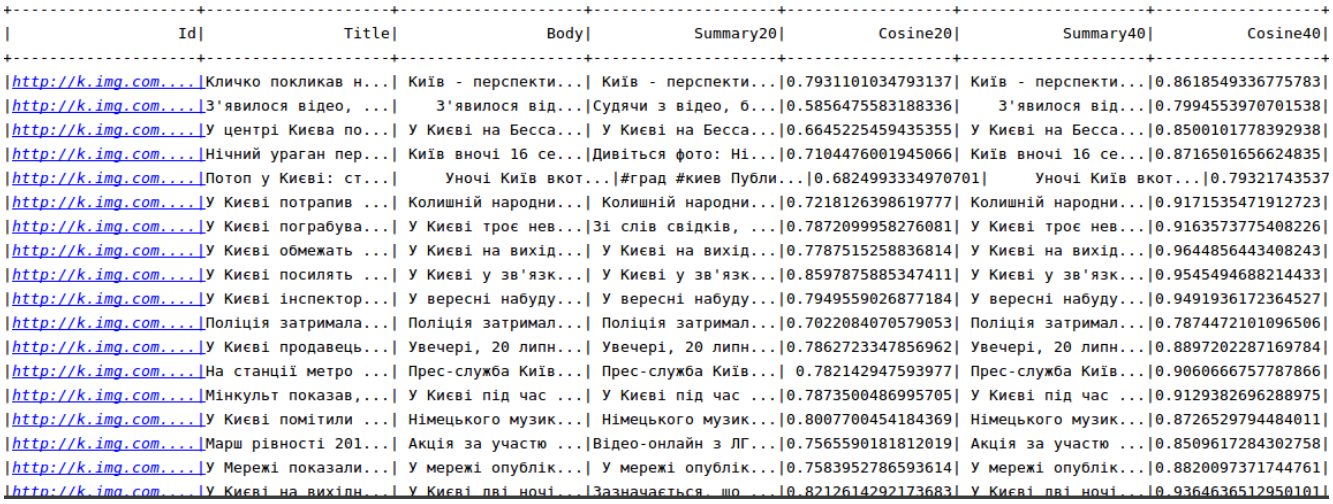
Для виміру швидкості обробки одного тексту був обраний середній за розміром текст об'ємом в 6609 слів. Обробка тексту включала в себе розбиття на речення та слова, лематизація та POS-аналіз слів, видалення шумів. В результаті очищення від шумів було знайдено та видалено 1650 часто вживаних слів. Результати обробки наведені у таблиці 4.1.

Таблиця 4.1. - Результати обробки текстового документу

Назва процесу	*Загальний час обробки, с*	*В середньому на одне слово, с*
Токенізація	2.94	0.00047
Нормалізація та POS-аналіз	85.9	0.013
Видалення стоп-слів	82.51	0.05
Загалом	171.35	0.063

4.3. Точністьвизначеннякосинусуподібності

Наступним критерієм оцінки стала точність роботи косинусу подібності. Для його розрахунку було взято розмічену вибірку новин з рефератами на 20 та 40 відсотків та вказаним косинусом подібності (рисунок 4.1).



Id	Title	Body	Summary20	Cosine20	Summary40	Cosine40
http://k.img.com....	Кличко покликав н...	Київ - перспекти...	Київ - перспекти...	0.7931101034793137	Київ - перспекти...	0.8618549336775783
http://k.img.com....	З'явилося відео, ...	З'явилося від...	Судячи з відео, б...	0.5856475583188336	З'явилося від...	0.7994553970701538
http://k.img.com....	У центрі Києва по...	У Києві на Бесса...	У Києві на Бесса...	0.6645225459435355	У Києві на Бесса...	0.8500101778392938
http://k.img.com....	Нічний ураган пер...	Київ вночі 16 се...	Дивіться фото: Ні...	0.7104476001945066	Київ вночі 16 се...	0.8716501656624835
http://k.img.com....	Потоп у Києві: ст...	Уночі Київ вкот...	#град #киев Публи...	0.6824993334970701	Уночі Київ вкот...	0.79321743537
http://k.img.com....	У Києві потрапив ...	Колишній народни...	Колишній народни...	0.7218126398619777	Колишній народни...	0.9171535471912723
http://k.img.com....	У Києві пограбува...	У Києві троє нев...	Зі слів свідків, ...	0.7872099958276081	У Києві троє нев...	0.9163573775408226
http://k.img.com....	У Києві обмежать ...	У Києві на вихід...	У Києві на вихід...	0.7787515258836814	У Києві на вихід...	0.9644856443408243
http://k.img.com....	У Києві посилять ...	У Києві у зв'язк...	У Києві у зв'язк...	0.8597875885347411	У Києві у зв'язк...	0.9545494688214433
http://k.img.com....	У Києві інспектор...	У вересні набуду...	У вересні набуду...	0.7949559026877184	У вересні набуду...	0.9491936172364527
http://k.img.com....	Поліція затримала...	Поліція затримал...	Поліція затримал...	0.7022084070579053	Поліція затримал...	0.7874472101096506
http://k.img.com....	У Києві продавець...	Увечері, 20 липн...	Увечері, 20 липн...	0.7862723347856962	Увечері, 20 липн...	0.8897202287169784
http://k.img.com....	На станції метро ...	Прес-служба Київ...	Прес-служба Київ...	0.782142947593977	Прес-служба Київ...	0.9060666757787866
http://k.img.com....	Мінкульт показав,...	У Києві під час ...	У Києві під час ...	0.7873500486995705	У Києві під час ...	0.9129382696288975
http://k.img.com....	У Києві помітили ...	Німецького музик...	Німецького музик...	0.8007700454184369	Німецького музик...	0.8726529794484011
http://k.img.com....	Марш рівності 201...	Акція за участю ...	Відео-онлайн з ЛГ...	0.7565590181812019	Акція за участю ...	0.8509617284302758
http://k.img.com....	У Мережі показали...	У мережі опублік...	У мережі опублік...	0.7583952786593614	У мережі опублік...	0.8820097371744761
http://k.img.com....	У Києві на вихідн...	У Києві дві ночі...	Зазначається, що ...	0.8212614292173683	У Києві дві ночі...	0.9364636512950101

Рисунок 4.1 - Вибірка для перерахунку косинусу подібності

Для кожного тексту з 1030 текстів було перераховано косинус подібності між початковим текстом та двома рефератами і порівняно з наведеними значеннями. Результати перевірки наведені в таблиці 4.2.

Таблиця 4.2. - Перевірка косинусу подібності

Порівнюваний текст	Середня абсолютна похибка	Максимальна абсолютна похибка	Середня відносна похибка
Реферат на 20%	0.0381	0.389	0.05
Реферат на 40%	0.0184	0.4	0.021

Як видно з таблиці середня абсолютна та відносна похибка досить малі, що дає змогу стверджувати про ефективність векторизації документів шляхом підрахунку TF-IDF міри.

Висновки до розділу

У розділі було проведено вимірювання основних параметрів бібліотеки: точності та швидкості. Проведено вимірювання швидкості попередньої обробки тексту. Враховуючи недосконалість апаратного забезпечення бібліотека показала достатньо хороший результат - 0.063 секунди на повну обробку слова. Також було визначено

точність підрахунку косинусу подібності та підтверджено ефективність обраного підходу до векторизації текстових документів.

5 РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

4.1. Описідеї проекту

Ціллю даного розділу є проведення аналізу стартап проекту, та визначення можливостей виходу на ринок продукту, що буде здатний конкурувати з вже існуючими рішеннями. Призначенням проекту є автоматизація процесів обробки текстової інформації для української та російської мов Сутністю розробки є створення інструменту автоматизованої обробки, що зможе проводити обробку в режимі реального часу та справитися з високою завантаженістю. Цільовою аудиторією для використання цього продукту є в першу чергу юридичні лиця, тобто компанії, яким необхідно проводити обробку та структуризацію текстових даних. Основною вигодою використання цього продукту є зменшення часу на виконання бізнес завдань пов'язаних з обробкою тексту. Для отримання цілісного уявлення про зміст ідеї, а також базові можливі потенційні ринки збуту, в межах яких потрібно шукати групи клієнтів, які в використовували даний продукт, опишемо переваги нашого продукту в таблиці 4.1.

Таблиця 4.1. - Переваги продукту

№ п/п	*Ідея*	*(потенційні) товари/концепції конкурентів*			*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*ApacheCore NLP*	*NLTK*			
1.	Виділення речень	Наявна	Наявна	Наявна		N	
2.	Токенізація	Наявна	Наявна	Наявна		N	
3.	Нормалізація	Наявна	Наявна	Наявна		N	

Продовження таблиці

№ п/п	*Ідея*	*(потенційні) товари/концепції конкурентів*			*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*ApacheCore NLP*	*NLTK*			
4.	Видалення стоп-слів	Наявна	Наявна	Наявна		N	
5.	POS-аналіз	Наявна	Наявна	Наявна		N	
6.	Сумаризація	Наявна	Відсутня	Відсутня			S
7.	Пошук подібних документів	Наявна	Відсутній	Відсутній			S
8.	Масштабованість	Наявна	Наявна	Відсутня			S
9.	Підтримка української мови	Наявна	Відсутня	Відсутня			S
10.	Підтримка російської мови	Наявна	Наявна	Відсутня			S
11.	Ціна	Платна	Безкоштовно	Безкоштовно	W		

Як видно з наведеної вище таблиці, можна зазначити, що проект має ряд переваг перед конкурентами, а саме: функції сумаризації та пошуку схожих текстів, можливість масштабування та роботи з великими даними, підтримка одночасно російської та української мов. Однак вагомим недоліком в очах потенційного користувача є те, що продукт платний. Проте ціна нашої бібліотеки повністю компенсується підтримкою українськомовних текстів, оскільки подібних послуг конкуренти не надають.

4.2. Технічний аудит проекту

Таблиця 4.2 – Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1	Виділення речень	Система розроблена на мові програмування Python 3.7	Наявна	Доступна
		Бібліотека langdetect	Наявна	Доступна
2	Токенізація	Система розроблена на мові програмування Python 3.7	Наявна	Доступна
		Бібліотека langdetect	Наявна	Доступна

Продовження таблиці

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
3	Нормалізація	Метод лематизації слів	Наявна	Доступна
		Бібліотека pymorphy2	Наявна	Доступна
4	Видалення стоп-слів	Словники частовживаних слів для української та російської мов	Наявна	Доступна
5	POS-аналіз	Бібліотека pymorphy2	Наявна	Доступна
6	Реферування	Екстрактивний метод TextRank	Наявна	Доступна
7	Знаходження подібності документів	Метрика TF-IDF	Наявна	Доступна
		Косинус подібності	Наявна	Доступна
8.	Можливість масштабування	Технологія ApacheSpark. Spark RDD, StructuredStream	Наявна	Доступна

Як видно з таблиці, проект, що розробляється в рамках дослідження, не потребує розробки нових унікальний технологій. А скоріше агрегує існуючі рішення та покращує їх.

4.3. Аналізринковихможливостейстартап-проекту

4.3.1. Аналіз попиту

Таблиця 4.3 – Попередня характеристика ринку

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	2
2	Загальний обсяг продаж, грн/ум.од	Невідомо
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	Невідома

4.3.2. Визначення потенційних груп клієнтів

Таблиця 4.4 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1	Структуризація текстових даних	юридичні особи, яким необхідно обробляти текстові дані в рамках реалізації бізнес-задач	готові заплатити високу ціну, за продукт, що вирішує їх проблеми	Автоматизація процесів, швидкість, масштабованість
2	Візуалізація текстового потоку	маркетологи та аналітики	необхідність у графічному інтерфейсі	Візуалізація даних. зручний інтерфейс

4.3.3. Аналіз ринкового середовища

Для аналізу ринкового середовища складемо таблиці факторів, що перешкоджають(таблиця 4.5) та сприяють(таблиця 4.6) ринковому впровадженню проекту. Фактори в таблиці відсортовані в порядку зменшення ваги.

Таблиця 4.5 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Персонал	Інтерфейс бібліотеки дещо відрізняється від існуючих аналогів	Написання детальної документації
2	Відмова у співпраці	Медична компанія відмовляється від співробітництва	Отримання коментарів щодо причини відмови, пропонування компромісу, введення змін

Таблиця 4.6 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	Науково-технічний	Вдосконалення інформаційної системи	Впровадження в роботу
2	Інноваційний	Підтримка української та російської мов	Покращення функцій для роботи з цими мовами
3	Економічний	Підтримка інновацій у виробництві	Підвищення / Пониження ціни на послугу

4.3.4. Аналіз пропозиції

Визначимо загальні риси конкуренції на ринку необхідні для аналізу пропозиції.

Таблиця 4.7 - Ступеневий аналіз конкуренції

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Тип конкуренції - чиста	В галузі є достатньо гравців, кожний з яких має свої сильні та слабкі сторони	Покращення існуючих послуг та впровадження інновацій
2. За рівнем конкурентної боротьби - міжнародний	Компанії конкуренти з різних країн	Максимальна локалізація продукту
3. За галузевою ознакою - глобальна	Продукт може використовуватись в будь-яких галузях	Інтеграція потреб різних галузей
4. Конкуренція за видами товарів: товарно-видова	Конкуренція між видами ПП, їх особливостями.	Вдосконалити ПП, враховуючи недоліки конкурентів
5. За характером конкурентних переваг - нецінова	Вдосконалення технології створення ПП, щоб собівартість була нижчою	Удосконалення моделі.
6. За інтенсивністю - марочна	Бренд присутній, але його роль незначна	Популяризація власної розробки шляхом участі в конференціях, семінарах

Висновок: За результатами проведеного аналізу було доведено, що існують можливості виходу продукту на ринок. В якості початкової стратегії можна обрати встановлення зв'язків з “вільними” клієнтами.

4.3.6. Обґрунтування переліку факторів конкурентоспроможності

Користуючись характеристиками ідей проекту та вимогами споживачів визначимо перелік факторів конкурентоспроможності(таблиця 5.8).

Таблиця 4.8 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1	Універсальність	Продукт поєднує у собі різні додаткові функції кожна з яких цікава різним споживчим сегментам
2	Масштабовність	Масштабованість продукту дозволяє використовувати його в роботі з великими даними
3	Швидкість	Порівняно висока швидкість роботи, що дозволяє користуватись функціями в режимі реального часу
4	Унікальність	Індивідуальний підхід до кожного клієнта з метою максимізувати його, а в результаті і свій прибуток.

4.3.7. Аналіз сильних та слабких сторін проекту

Таблиця 4.9 – Порівняльний аналіз сильних та слабких сторін spark_lp

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з CoreNLP*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Універсальність	15					+		
2	Масштабованість	15				+			
3	Швидкість	10				+			
4	Унікальність	15						+	

4.3.8. SWOT-аналіз

Завершальним етапом ринкового аналізу можливостей реалізації проекту є складання SWOT (матриця аналізу сили та слабкості), аналізу загроз (проблем) та можливостей (таблиця 4.10) на основі обраних ринкових загроз та можливостей, а також сильні та слабкі сторони. слабкі сторони (таблиця 4.9).

Перелік загроз та ринкових можливостей базується на аналізі загроз та факторів маркетингового середовища. Ринкові загрози та ринкові можливості - це наслідки (очікувані результати) впливу факторів і, навпаки, вони ще не реалізовані на ринку та мають певну ймовірність реалізації. Наприклад: падіння доходів потенційних споживачів - фактор загрози, на основі якого можна скласти прогноз підвищення значення цінового фактора у виборі товару і, отже, - цінової конкуренції (а це є - ринкова загроза).

Таблиця 4.10 – SWOT- аналіз стартап-проекту

Сильні сторони: Унікальність, масштабованість, швидкість роботи	Слабкі сторони: можливе нерозуміння потреб ринку
Можливості: Розширення споживчого сегменту, підвищення ціни	Загрози: поява нових конкурентів

4.3.9. Альтернативи поведінки

На основі SWOT-аналізу ми розробимо альтернативи поведінці на ринку (перелік заходів) для виведення на ринок стартового проекту та приблизний оптимальний час їх реалізації на ринку, враховуючи потенційних проектів-конкурентів, які можна вивести на ринок. Визначені альтернативи аналізуються з точки зору часу та ймовірності отримання ресурсів.

Таблиця 4.11 – Альтернативи ринкової поведінки

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	Активна реклама у сферах та засобах комунікації потенційних клієнтів	Висока за рахунок того, що продукт буде популяризуватись.	1 рік
2	Пробний період	Висока за рахунок того, що у клієнта з’явиться бажання купити ліцензію на довгий термін після закінчення пробного періоду	1.5 роки

В якості альтернативи варто обрати комбінацію наведених варіантів; активне просування разом з введенням пробного періоду, що забезпечить максимальну віддачу.

4.4. Розробленняринковоїстратегії проекту

4.4.1 Опис цільових груп потенційних користувачів

Таблиця 4.12 – Цільові групи

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1	Компанії, яким необхідні структуровані дані	Готові, оскільки продукт значно спростить їх роботу	Високий	Середня, небагато компаній має власну систему обробки а аналоги серед конкурентів мають недоліки	Просто, оскільки попит великий, а конкуренція досить низька

Продовження таблиці

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
2	Маркетологи та аналітики	Готові розглянути як варіант	Середній	Середня, існують варіанти, що розробленні для подібних цілей, тож потрібно буде переконати їх в перевазі саме нашого продукту	Середня складність, через наявність конкуренції та помірний попит

За результатами аналізу потенційних груп споживачів були обрані цільові групи - 1 і 2, для яких буде пропонуватися даний товар, та стратегія захоплення ринку - диференційована маркетингова стратегія (компанія працює з різними сегментами).

4.4.2. Базова стратегія розвитку

Щоб почати роботу у вибраних сегментах ринку варто спочатку сформувати базову стратегію розвитку.

Таблиця 4.13 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
	Орієнтація на вітчизняних іноземних інвесторів	Ставка на універсальність продукту, та відносно високу якість. Стратегія товарної спеціалізації - пропонування товару(з певними модифікаціями) різним групам ринку.	Функціонал. Універсальність.	Стратегія диференціації - охоплення декількох сегментів ринку й розроблення для кожного з них окремого комплексу маркетингу

Отже, основною стратегією є вибір стратегії диференціації на основі потреб користувачів. Альтернативою ж буде орієнтація на вітчизняних інвесторів.

4.4.3. Вибір стратегії конкурентної поведінки

Наступним логічним кроком є обрання базової стратегії розвитку конкурентної поведінки, а саме: орієнтація на “вільних” та “зайнятих” клієнтів, запозичення характеристик конкурентів.

Таблиця 4.14 – Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки**
	Ні	70% шукати нових клієнтів, 30% — переконувати інших клієнтів скористатись саме нашими послугами	В основу продукту лягла звичайна бібліотека для обробки текстових даних яких є досить багато, унікальність полягає в підтримці української та російської мов та спробі зробити цю систему універсальною.	Фланговий наступ - спочатку забрати вільних клієнтів, потім почати роботу з клієнтами конкурентів.

4.4.4. Стратегія позиціонування

На основі потреб споживачів від вибраних сегментів до постачальника (стартап-компанії) та товару, а також обраної базової стратегії розвитку та стратегії конкурентної поведінки розробляється стратегія позиціонування, яка полягає у формуванні ринкової позиції (група асоціацій) за якою користувачі матимуть змогу безпомилково визначити бренд / проект.

Таблиця 4.15 – Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
1	Швидкість	Стратегія диференціації	Достатньо висока швидкість	Відчуття якості продукту
2	Масштабованість	Стратегія диференціації	Можливість обробки великих даних	Відчуття надійності продукту
3	Універсальність	Стратегія диференціації	Продукт може вирішувати декілька задач на відміну від звичайних текстових аналізаторів	Відчуття повноти функціоналу, зручності

4.5. Розробленнямаркетинговоїпрограмистартап-проекту

4.5.1. Маркетингова концепція товару

Першим кроком є формування маркетингової концепції товару, який отримує споживач. Для цього нам потрібно узагальнити результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.16 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1	Масштабовність	Надає можливість роботи з великими даними, що дозволяє інтегрувати продукт для компанії будь-якого рівня.	Обробка великих даних та потоків текстових даних.

Продовження таблиці

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
2	Швидка робота	Комфортна швидкість навіть при роботі з порівняно великими об’ємами даних. Швидка інтеграція з сайтом/сервісом/додатком клієнта.	Висока швидкість інтеграції та роботи.

4.5.2. Маркетингова модель товару

Ми розробимо трирівневу маркетингову модель товару: уточнення ідеї товару та / або послуги, її фізичних складових, характеристик процесу її постачання.

Таблиця 4.17 – Опис трьох рівнів моделі товару

<table>
<tr><th>Рівні товару</th><th colspan="3">Сутність та складові</th></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Програмне забезпечення адаптивне, та може бути застосоване у будь якій сфері. Висока швидкість роботи та масштабованість забезпечується хмарними технологіями.</td></tr>
<tr><td rowspan="6">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Вартість обслуговування
2. Знижки
3. Безвідмовність
4. Собівартість</td><td>М
Нм
М
М</td><td>Вр
Вр
Тх
Тх</td></tr>
<tr><td colspan="3">Якість: стандарти - PEP 8
Тестування - unittest/ integrationtest</td></tr>
<tr><td colspan="3">Пакування програмна бібліотека</td></tr>
<tr><td colspan="3">Марка:</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу - безкоштовна ліцензія сроком на 1 місяць</td></tr>
<tr><td colspan="3">Після продажу - безкоштовні оновлення та модифікація продукту. Допомога з налаштуванням роботи за окрему плату.</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання:
Патенту на продукт, унікальності деяких деталей та комерційної таємниці.</td></tr>
</table>

4.5.3. Визначення меж встановлення ціни

Наступним кроком є визначення цінових меж, якими слід керуватися при визначенні ціни потенційного продукту (кінцева ціна визначається під час фінансово-економічного аналізу проекту), що передбачає аналіз ціни на подібні або замінні товари, а також аналіз доходу цільової групи споживачів. Аналіз проводиться за допомогою експертного методу.

Таблиця 4.18 – Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
1	60 000 - 1 000 000 грн (собівартість розробки текстового аналізатора під потреби компанії)	Відсутні	від 50 000 грн/місяць	від 5 000 грн на місяць і вище. Верхня ціна договірна і залежить від потреб та обсягу використання.

4.5.4 Оптимальна система збуту

Наступним кроком є визначення оптимальної системи продажів, за якою приймається рішення: здійснювати власні продажі або залучати сторонніх посередників (власна або задіяна система продажів); вибір та обґрунтування оптимальної глибини каналу збуту; вибір та обґрунтування типу посередників.

Таблиця 5.20 - Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1	Пробний період, купівля ліцензії на використання	Контакти зі споживачами, формування попиту, досліди в області маркетингу	Канал нульового рівня (виробник безпосередньо продає товару клієнту)	Ліцензія на використання

4.5.5 Розроблення стратегії маркетингових комунікацій

Останньою складовою маркетингової програми є розробка концепції маркетингової комунікації, заснованої на заздалегідь обраній основі позиціонування, на визначених специфікаціях поведінки клієнтів.

Таблиця 4.20 – Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
1	Зазвичай сезонами коли проводиться 80% всіх заходів є осінь та зима, решта 20% припадає на весну та літо.	Соцмережі, пошта, конференції, сповіщення	Комплексний підхід	Повідомлення має переконати клієнта, що використання продукту принесе йому прибуток	“Підкорюємо інтернет”

ВИСНОВКИ

В рамках магістерської дисертації розроблено бібліотеку обробки текстової інформації для ApacheSpark. Програмний продукт виконаний на мові програмування Python у вигляді бібліотеки, яка містить методи для попередньої обробки текстових даних, векторизації документів, пошуку міри схожості документів, та реферування. Також особливістю бібліотеки є підтримка роботи з потоками даних.

У першому розділі розглянуто етапи обробки тексту та нюанси, які зустрічаються на кожному з етапів. Описано варіанти векторизації документів та підходи до реферування тексту. Наведено порівняння існуючих рішень для обробки текстових даних і виділено переваги нашого продукту. Розписано завдання, які мають бути вирішені в рамках дослідження.

У другому розділі було детально описано сервіси, що можуть бути використані для масштабування бібліотеки та роботи з потоками. Наведено базову архітектуру для кожного рішення та опис їх переваг та недоліків. Результатом розділу є обгрунтування чому було обрано саме ApacheSpark.

У третьому розділі розглянуто архітектуру програмного забезпечення та варіанти його використання в проектах. Наведено приклад рішення для архітектури Kafka + Elasticsearch + Kibana.

Для визначення ефективності роботи програмного забезпечення було виміряно швидкість його роботи на достатньо великому наборі даних. Також проведено реферування текстів та вирахувано косинус подібності між текстом та рефератом.

У п'ятому розділі було проведено аналіз ринкових можливостей запуску стартап-проекту, розроблено стратегію його розвитку та просування, виділено цільові групи клієнтів, створено модель поведінки з конкурентами та розроблено маркетингову програму проекту.

Наукова новизна розробки полягає у створенні програмної бібліотеки для технології ApacheSpark на мові Python, яка на відміну від існуючих містить

функції реферування та підтримку обробки українськомовних текстів.

Усі поставленні задачі наукової роботи були виконані. Було створено програмну бібліотеку з усіма методами необхідними для обробки текстової інформації, її структуризації та векторизації. Також було додано підтримку для методів реферування текстів і проведено аналіз ефективності роботи бібліотеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Silvana C. Exploratoryanalysisoftextualdatastreams / C. Silvana, F. Alfio, S. Stefano. // FutureGenerationComputerSystems. – 2017. – №68. – p. 391–406.

2) KaziSaidulHasan, VincentNg. AutomaticKeyphraseExtraction: A SurveyoftheStateoftheArt. UniversityofTexasatDallas, HumanLanguageTechnologyResearchInstitute.

3) Недільченко, О. С. Етапи та методи автоматичного вилучення ключових слів / О. С. Недільченко. – журнал "Молодий вчений". – 2017. – № 22. – с. 60-62.

4) Л.Е. Чалая, Ю.Ю. Харитонова Метод векторно-графовоїкластеризації документів в системах обробки текстової інформації

5) Mikolov T., Chen K., Corrado G., Dean J. EfficientEstimationof Word RepresentationsinVectorSpace / InProceedingsofWorkshopat ICLR. – 2013.

6) KaziSaidulHasan, VincentNg. AutomaticKeyphraseExtraction: A SurveyoftheStateoftheArt. UniversityofTexasatDallas, HumanLanguageTechnologyResearchInstitute.

7) RadaMihalcea, PaulTarau. TextRank: BringingOrderintoTexts, UniversityofNorthTexas, 2004. Режим доступу: http://www.aclweb.org/anthology/W04–3252

8) NanMaa, JianchengGuan, YiZhaoc. BringingPageRanktothecitationanalysis. // InformationProcessing&Management. – 2008. – №44. – p. 800-810.

9) Ukrainian NLP LibraryforApacheSpark.URL: https://github.com/oliyura/UANLP/ [Назва з екрана]

10) Korobov, M. (2015, April). MorphologicalanalyzerandgeneratorforRussianandUkrainianlanguages. InInternationalConferenceonAnalysisofImages, SocialNetworksandTexts (pp. 320-332). Springer, Cham.

11) Аршакян Г.Д. Олійник Ю.О. Огляд підходів та методів автоматичного реферування тексту. Інформаційні системи та технології управління: матеріали всеукр. наук.-практ. конф. молодих вчених та студентів, (м. Київ, 26 лист. 2019 р.). Київ, 2019. С. 44-48.

12) VishalGupta , Gurpreet S. Lehal «A SurveyofTextMiningTechniquesandApplications» JournalofEmergingtechnologiesinwebintelligence, vol,1 no1 August 2009.

13) ApacheStorm [Електронний ресурс] // 2019 – Режим доступу до ресурсу: https://storm.apache.org/.

14) ApacheHadoop [Електронний ресурс] – Режим доступу до ресурсу: https://hadoop.apache.org/.

15) AmazonKinesis [Електронний ресурс] – Режим доступу до ресурсу: https://aws.amazon.com/ru/kinesis/.

16) ApacheSpark [Електронний ресурс] – Режим доступу до ресурсу: https://spark.apache.org/.

17) Burgess, C., Livesay, K., &Lund, K. (1998). Explorationsincontextspace: Words, sentences, discourse. DiscourseProcesses, 25, 211-257.